AMIGA DOS

DAS MAGAZIN FÜR AKTIVE AMIGA-ANWENDER

DM 6,50
sfr 6,50
öS 52

2/90

AMIGA DOS

2/90
ISSN 0937-2717
DMV-Verlag

CONTEXT-AMIGA
Drucken im Multitasking-Betrieb

Der Videotoaster
DAS Superding

JEDE MENGE TIPS & TRICKS
- Sprite-Editor mit 16 Farben
- Logische Devices - alles Logo?
- Flugjoystick-Bauanleitung
- Guru Off - Meditations leicht und verständlich
- und, und, und...

Hardware
Drucker im Test
Eprom-Brenner VESUV

Grafik
3D-Rotationskörper

MIT GROSSEM SPIELE-TEIL

GFA für AMIGA

GFA-BASIC 3.0 Interpreter Amiga

Strukturiertes Programmieren, über 300 Befehle und Funktionen, zusätzlich alle Betriebssystemfunktionen, extrem hohe Geschwindigkeit. **DM 198,–**

neu

GFA-BASIC 3.0 Compiler Amiga

Mit dem integrativen Compiler werden Ihre GFA-BASIC-Programme noch schneller. Viele Optionen und Linker (kompatibel zu A-Link und B-Link) für andere Programmiersprachen im Lieferumfang enthalten. **DM 99,–**

neu



Der Einstieg in GFA-BASIC 3.0 Amiga

Ein Lehrbuch für Programmieranfänger. Dietmar Schell vermittelt auch dem unerfahrenen Programmierer Ideen und Anwendungsbeispiele für das Programmieren in GFA-BASIC. 248 Seiten, Hardcover, ISBN 3-89317-009-X **DM 29,–**

neu



Training für Fortgeschrittene GFA-BASIC 3.0

Wer schon Erfahrung auf dem Amiga oder in irgendeinem BASIC-Dialekt hat, wird von den beiden Autoren bestens betreut. Man erfährt und lernt eine Menge über Programmiertricks, nützliche und verwendbare Prozeduren, Anwendungen und die Besonderheiten des GFA-BASIC für Amiga. 329 Seiten, Hardcover, inkl. Diskette, ISBN 3-89317-010-3 **DM 49,–**

neu

GFA-ASSEMBLER Amiga

Professioneller Makro-Assembler für 68000-Programmierer: Leistungsfähiger Editor mit integriertem Assembler und Linker. Nachladbarer Debugger. Jetzt auch für die Commodore-Amiga-Computer lieferbar. **DM 149,–**

neu

ZOETROPE

Das Computer-Animationssystem für Ihren Amiga mit der Funktionalität und den Eigenschaften, die man nur bei erstklassigen Grafiksystemen findet. Das professionelle 2D-Animationsprogramm von ANTIC-Software, exklusiv von GFA. Umfangreiches Handbuch und Programm in Deutsch. **DM 198,–**

Anruf genügt
0211/5504-0

GFA Systemtechnik GmbH
Heerdter Sandberg 30-32
D-4000 Düsseldorf 11
Tel. 0211/5504-0 · Fax 0211/550444





AMIGA mon amour...

...wird so mancher denken, der seinen frisch erworbenen AMIGA stolz nach Hause trägt. Grafik und Sound vom Feinsten, dazu eine grafische Benutzeroberfläche, die genauere Kenntnisse der Maschine auf ein Minimum reduziert.
Wie elegant und mühelos sich doch Raytracing-Grafiken auf dem Monitor bewegen... Kann man den Angaben der Hersteller Glauben schenken, ist fast jedermann ohne Vorkenntnisse in der Lage, solche Kunstwerke und Animationen flugs auf den Bildschirm zu zaubern.
Hat man sich an den großzügig bemessenen Demos sattgesehen, keimt natürlich der Wunsch auf, "mal eben" solche beeindruckenden Grafiken selbst zu erstellen.
Spätestens auf Seite 272 des in englischer Sprache gehaltenen Raytracing-Handbuches kommt dann der erste Verdacht auf, daß vielleicht doch etwas mehr als ein AMIGA in der Grundausstattung und ein Anwendungsprogramm notwendig sind, um Grafik in solcher Qualität auf den Bildschirm zu zaubern.
Ein Buch muß her, das die wesentlichen Fragen in Kürze klärt. Nach gründlichem Studium dieses Buches sind mehr neue Fragen aufgeworfen, als bestehende beantwortet. Wie war das doch gleich mit diesen ominösen Key-Frames, Cells und Editorscreens? Und daß manche Anwendungsprogramme rigoros ein volles MByte beanspruchen, um überhaupt arbeiten zu können, wirft die Frage nach zusätzlicher Hardware auf.
Der Gang zum Hardwarehändler endet dann meist in völliger Verwirrung. 2 MBytes, Zweitlaufwerk oder gleich eine Harddisk anschaffen?
Sind alle Voraussetzungen geschaffen, steht dem AMIGA-Glück nichts mehr im Wege. Oder doch? Die Anleitung zum Einrichten einer Datadisk behandelt diesen Themenbereich mit dem lapidaren Satz: Achten Sie darauf, rechtzeitig eine Datadisk einzurichten. Um Speicherplatz zu sparen, empfehlen wir, eine Diskette ohne Icons und Trashcan zu verwenden. Ein Blick ins Handbuch offenbart leider nur sehr dürftige Beschreibungen zur Anwendung der DOS-Befehle. Diesem Eindruck wird sich so mancher frischgebackene Amiganer nicht verschließen können.
Und hat sich die anfängliche Euphorie und der darauf folgende Katzenjammer ein wenig gelegt, macht so mancher Amiganer die Feststellung, daß auch BASIC durchaus seine Reize haben kann. Oder sollte es lieber doch gleich ASSEMBLER sein?
Was fehlt, ist ein Scout durch den Soft- und Hardware-Dschungel, der zu all diesen Fragen gleichermaßen Stellung nimmt. Und Workshops zu den am weitesten verbreiteten Grafikprogrammen und Programmiersprachen können die Kosten bei der Literaturbeschaffung in einem vernünftigen Rahmen halten.
Doch wo findet man derart kompakt aufbereitetes Wissen?
Natürlich in der AMIGA DOS, deren Februar-Ausgabe Sie gerade in den Händen halten. Und sollte das nicht genügen – kompetente Hilfe ist nur einen Anruf entfernt. Beziehungskisten, gerade mit Computern, sind eben nicht immer einfach. Und damit Sie nicht den Eindruck bekommen, sich einen Hausdrachen angelacht zu haben, werfen Sie einen Blick in die neue AMIGA DOS. Es lohnt sich.

Herzlichst Ihr

Markus Malejka

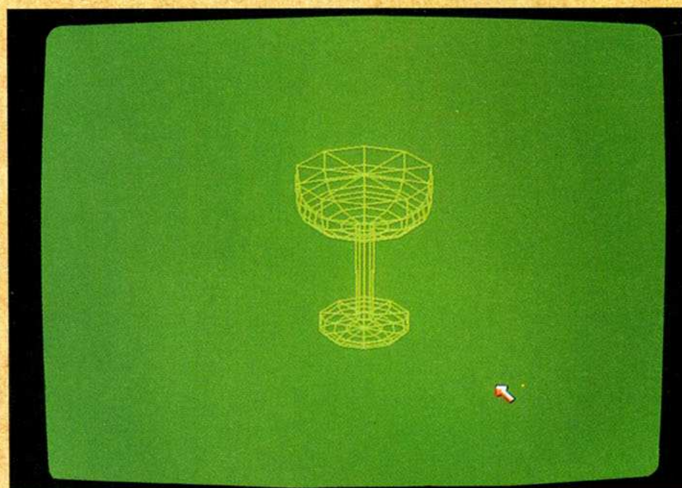
NEWS & TRENDS

SOFTWARE

HARDWARE

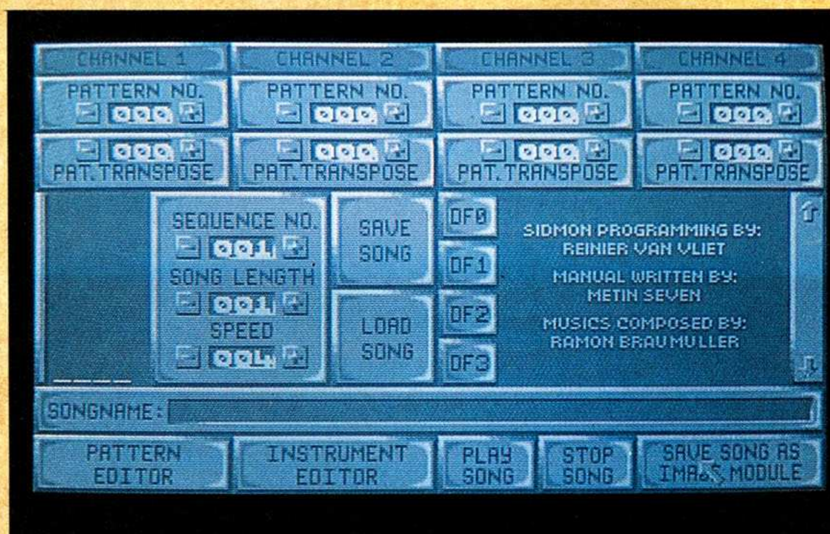
TIPS & TRICKS

KURS



3D-Rotationskörper leicht erstellt. Mit diesem Programm kann fast jeder dreidimensionale Körper dargestellt werden. Komplett per Maus gesteuert, ist es einfach zu bedienen und überzeugt durch seine Leistungsfähigkeit. Überzeugen Sie sich selbst

Seite 35



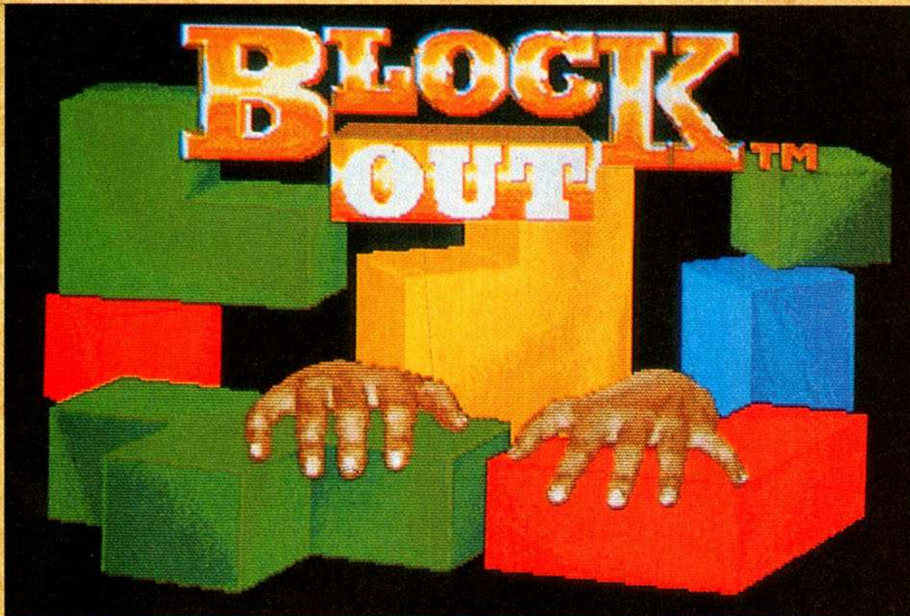
Mit Sidmon steht ein neuer Musikeditor ins Haus, der durch sein Bedienerfreundlichkeit und seine Leistungsdaten neue Maßstäbe setzen soll. Kann Sidmon diesem Anspruch gerecht werden? Lesen Sie selbst auf

Seite 24

Drucker für den Amiga gibt es wie Sand am Meer. Doch welchen Drucker für welchen Zweck? Wir haben uns für Sie einmal auf dem Markt umgesehen, um Ihnen einen Überblick zu verschaffen. Unsere Marktübersicht zeigt die wesentlichen Fakten auf einen Blick

Seite 93





Block out setzt Tetris in die dritte Dimension. Nun reicht es nicht mehr, Linien zu bilden, geschlossene Flächen sind hier gefragt. Keine einfache Aufgabe, wenn Sie es selbst einmal versuchen wollen

Seite 128



Grundlagen Grafik und Animation: das richtige System bei der Arbeit ist ebenso wichtig wie Grundinformationen zur Animationserstellung. AMIGA DOS ist den Geheimnissen der Animation auf den Grund gegangen. Näheres auf

Seite 18

Zu den Sternen und darüber hinaus

Wer immer schon einmal die Weiten der Galaxis erkunden wollte, konnte bisher mit seinem Amiga Atemberaubendes erleben.

Von Halley Project bis Elite reicht das Spektrum der Weltraum-Erkundungsspiele. Electronic Arts legt nun einen neuen Vertreter des Genres vor, der auf MS-DOS-Rechnern schon für Furore sorgte. Wer allerdings eine knallharte Raumflugsimulation erwartet, wird enttäuscht werden. Starflight geht die Sache eher wie ein Rollenspiel an und versucht, einen spielbaren, unterhaltsamen Mittelweg zu finden. Mit einer Mannschaft von sechs Spezialisten gilt es, eine Galaxie mit 270 Sternsystemen und über 800 Planeten zu erkunden. Neben vielen verschiedenen außerirdischen Rassen werden dem Spieler auf seiner Reise durch das All viele kosmische Sehenswürdigkeiten begegnen.



Produktfoto: Starflight

Neue Tastatur für den Amiga 2000

Durch das ständige Gehämmer auf die armen Tasten des Keyboards wird die Funktionsfähigkeit der Tastatur drastisch dezimiert. Kommt es eventuell auch noch zu Zwischenfällen, wie ungewollter Tastaturnässe (z.B. durch Ausschütten von Kaffeetassen), können Sie Ihr Keyboard sicherlich irgendwann einmal abschreiben. Eine neue Tastatur muß her!

Eine neue Tastatur für den Amiga 2000 könnte beispielsweise die neue Tastatur Alphakey der Firma AFC sein. Das Besondere an dieser Tastatur ist ein austauschbares Steckmodul, in dem Standardbelegungen abgespeichert werden können. Diese Tastenbelegungen bleiben automatisch in dem Modul gespeichert, wobei kein Einfluß auf die Hard- bzw. Software genommen wird. Ganze Befehlssequenzen mit programmübergreifenden Kommandos lassen sich auf eine Taste legen (man bedenke die komplexen Compiler- und Linkeraufrufe). Weiterhin ist ein Passwortschutz über die Tastatur möglich. Hochwertige Hard- und Software hat ihren Preis, so auch Alphakey: Die Tastatur ist für 1490,– DM zu haben.

Bezugsquelle:
AFC GmbH
Theodor-Heuss-Ring 52
5000 Köln 1
Tel.: 0221/13 40 36
Fax: 0221/13 29 00

Blendschutzfilter für Datenmonitore

Die Kathodenstrahlröhren der Bildschirme arbeiten mit Hochspannung. Als unerwünschter Nebeneffekt baut sich zwischen dem Bediener und dem Bildschirm ein elektrostatisches Hochspannungsfeld von 20.000 Volt und mehr auf.

Durch diese hohe Spannung werden winzige Schwebeteilchen wie Staub, Rauch, Ruß etc. in Bewegung gesetzt. Sie prallen auf Haut und Augen und können Reizungen hervorrufen. Von Fachleuten wurden daher Bildschirmfilter entwickelt, die das Arbeiten vor Monitoren sicherer gestalten sollen. Informationen über Blendschutzfilter sowie die Filter selbst können bei nachstehender Adresse bezogen werden.

Bezugsquelle:
Langkavel GmbH
Langwiese 15
6457 Maintal 2
Postfach 20 02 28
Tel.: 0 61 09/ 6 79 20

Mehr Farben auf dem Amiga

Die englische Firma HiTension Computer Services Limited kündigten eine neue Grafikerweiterungskarte für den Amiga 2000 an. Die Karte namens Amigraphex, die in zwei Versionen auf den Markt kommen soll, belegt einen internen Expansion-Slot. Mit der ersten Version von Amigraphex ist eine Bilddarstellung von bis zu 1024x768 Pixel Auflösung möglich. Ferner sind 256 Farben aus einer Farbpalette von 16 Millionen Farben darstellbar. Der Prozessor ist mit einer Taktfrequenz von 50 MHz ausgestattet.

Amigraphex II ist mehr für CAD- und DTP-Anwendungsprogrammen gedacht. Die Auflösung dieser Karte liegt bei 1280x1024, wobei die gleiche Farbpalette wie bei der "kleinen" Version zur Verfügung steht. Der Preis ist bei 899 englischen Pfund (Amigraphex I) bzw. 1599 Pfund (Amigraphex II) angesetzt.

Bezugsquelle:
HiTension Computer Services Ltd.
Alexander House, Station Road
Aldershot, Hampshire GU11 1BQ
Tel.: (0044) 0252 344454
Fax: (0044) 0252 23417



Bildvorlage: Blendschutzfilter

Leichter lernen – Teachware auf dem Amiga



Produktfoto: Heureka Teachware

Daß der Amiga auch zum Lernen eingesetzt werden kann, beweisen viele Programme der sogenannten Teachware. Neben Fremdsprachen können auch mathematische Grundlagen mit der Heureka-Teachware erlernt werden. Der Lerneffekt wird dabei auf spielerische Weise nähergebracht. Wir haben uns in der Redaktion einen Einblick in ein mathematisches Kurvendiskussions- und ein Englisch-Lernprogramm verschafft. Bei Zenon geht es um mathematische Grundlagen für die Oberstufe des Gymnasiums. Hier können unter anderem Formeln eingegeben, deren erste und zweite Ableitungen und Tangenten bzw. Asymptoten grafisch auf den Bildschirm und/oder auf einen angeschlossenen Drucker gebracht werden. Der Einfachheit in der Bedienung halber ist das Programm mit übersichtlichen Pulldown-Menüs ausgestattet.

Learning English Modern Course ist ein Englischkurs, bei dem Falscheingaben präzise angezeigt und sofort korrigiert werden können. Das Besondere an diesem Programm sind die verschiedenen Abfragevarianten, die sich vom Benutzer einstellen lassen.

Bezugsquelle:
Ostermann Verlag
Paul-Hösch-Str. 4
8000 München 60
Tel.: 089/82 01 20 0

Deluxe Video III, auf dem Weg zum Anwender

Mit Deluxe Video legte Electronic Arts eines der ersten Programme vor, die den Anwender in die Lage versetzen, Grafik, Musik und Animation zu kompletten Video-Clips zu verbinden. Nun soll mit der Version 3.0 des Produktes eine neue Ära des Computervideos auf dem Amiga anbrechen. Deluxe Video III unterstützt alle Amiga-Bildschirm-Modi (auch HAM, Hires und Interlace). Weitere angekündigte Features sind schnelle Animationsroutinen und Musikausgabe über Midi. Deluxe Video zeigt insbesondere in Zusammenarbeit mit Deluxe Paint III seine Fähigkeiten. Deluxe Video unterstützt auch Overscan, so daß erstellte Clips auch auf einen Videorekorder überspielt werden können, ohne daß bei der Vorführung störende Ränder zu sehen sind. Das Programm wird auf allen Amigas mit mindestens 1 MByte RAM laufen und auch eine Harddisk unterstützen. Auch für die Weiterverwendung erstellter Videos ist durch ein frei kopierbares Playerprogramm gesorgt. Registrierte Benutzer von Deluxe Video I + II können einen Update-Service in Anspruch nehmen.

Pro Access V2.17 – Nachlese

Das Freezer-Modul Pro Access (wir berichteten in der letzten Ausgabe) ist um einige Dinge erweitert worden. Der Bremsbereich der eingebauten Prozessorbremse wurde vergrößert; die Verarbeitungsgeschwindigkeit des Prozessors wird nun von 71% bis auf 1% seiner ursprünglichen Geschwindigkeit gebremst. Auch beim Einschalten wurden Verbesserungen angebracht: So erscheint jetzt beim Bootvorgang ein Titelbild, das das korrekte Arbeiten von Pro Access anzeigt. Der Zeichensatz, der eher an 64er-Zeiten erinnert, wurde nun durch einen Topaz/8-ähnlichen Font ersetzt, um die Lesbarkeit zu verbessern.

Wer!Was!Wann!Wo!



Screenshot: Wer!Was!Wann!Wo!

Sich in der heutigen Welt mit all dem Chaos und der Hektik zurechtzufinden, ist nicht ganz leicht. Termine, Notizen, Verabredungen... wer kann das alles im Kopf behalten? Berge von Papier auf dem Schreibtisch, unvollständige oder veraltete Telefonlisten, verlegte Terminkalender, und der Hund sitzt mittendrin, wer möchte da nicht verzweifeln? Um dieses Manko aus dem Weg zu räumen, existiert ein Softwareprodukt namens Wer!Was!Wann!Wo!. Es handelt sich dabei um ein speicherresidentes Programm, das den Benutzeralltag organisieren soll. Es verwaltet die Informationen, wobei für jede Person eine Karteikarte mit den Eintragungen ähnlich eines Telefon- oder Adreßbuches zur Verfügung steht. Für die Übersicht in der Terminplanung ist auch gesorgt: Jeder Termin ist zu einer bestimmten Zeit und gewöhnlich auch mit einer bestimmten Person eingetragen. Wer!Was!Wann!Wo! enthält eine Alarm-Uhr, die den Benutzer durch Blinken und/oder durch einen Ton an einen Termin erinnern kann. Es ist sehr benutzerfreundlich gestaltet und unterstützt auch Modems. Mit Hilfe dieser Option ist es möglich, Telefonnummern vom Computer wählen zu lassen. Weiterhin können die Termine auch auf einem angeschlossenen Drukker ausgegeben werden. Besonderheit bei diesem Programm: zum Schreiben von Serienbriefen mit einer Textverarbeitung können die Adressen von Wer!Was! WannWo! eingebunden werden. Der Preis liegt bei zirka 150,– DM.

Bezugsquellen:
Schweiz:
Microtron
Postfach 69
Bahnhofstr.2
CH-2542 Pieterlen
Tel.: 032/872429

Deutschland:
DTM Werbung & EDV
Poststr. 25
6200 Wiesbaden-Bierstadt
Tel.: 06121/560084

Die neue Video-Dimension

Amiga und Video – ein Gespann, das sowohl in professionellen Filmstudios als auch im Privatbereich immer mehr Anklang findet. Dabei kommen immer leistungsfähigere Programme auf den Markt, die dieses Duo unterstützen. Ein Produkt, das in diese Bresche schlägt, liegt mit VES[one] zugrunde. VES[one] ist ein professionelles Video-Nachbearbeitungssystem für den privaten Anwender. Vier Formen der Nachbearbeitung von Videos sind in VES[one] integriert.

Ein Titelgenerator ermöglicht, auf einfachste Weise Lauf-, Unter- und Effekttitel zu erzeugen. Hierbei können Texte mit bis zu acht verschiedenen Farben (aus 4096) verwendet, deren Laufgeschwindigkeit stufenlos variiert und mit 18 unterschiedlichen Wisch- und Blendeffekten versehen werden. Standardmäßig wird die Software mit vier verschiedenen Fonts und vier unterschiedlichen Schriftgrößen ausgeliefert.

Der eingebaute Video-Digitizer – Digi View Gold – schafft die Basis für digitale Bildverarbeitung. Die digitalisierten Videovorlagen können in vielfältiger Art in Farbe und Form durch den Amiga verändert werden. Das Ergebnis kann dann abgespeichert und zu 2D- bzw. 3D-Trickanimationen weiterverarbeitet werden. Die Schaltzentrale von VES[one] ist der integrierte Videomischer. Hier laufen die Video- und Amiga-Signale zusammen und können mit verschiedenen Effekten kombiniert werden. Neben dem Ein- und Ausblenden von Amiga-Titelgrafiken in das laufende Videosignal besteht die Möglichkeit, aus drei verschiedenen Wischeffekten Trickeinblendungen vorzunehmen.

Trotz des hohen technischen Standards moderner Videokameras sind Über- und Unterbelichtungen bzw. Farbsättigungsprobleme nicht auszuschließen. In diesen Fällen tritt die vierte Komponente von VES[one], der Videosignalprozessor, in Aktion, indem er eine Beeinflussung von Kontrast, Helligkeit und Farbsättigung ermöglicht. So lassen sich auch auftretende Kopierverluste beim Überspielen auf ein Minimum reduzieren. Der Amigasignalprozessor ermöglicht darüber hinaus noch eine Anpassung des Amiga-Signals an das Videosignal.

Die Software Videopage benötigt mindestens 1 MByte RAM, die Anleitungen für Hard- und Software sind in Deutsch gehalten. Der Preis liegt bei knapp 2700,– DM.

Bezugsquelle:
Videocomp
Berner Str. 17
6000 Frankfurt 56
Tel.: 069/5 07 69 69
Fax: 069/5 07 62 00



Bildvorlage: VES[ONE]

Neuer Assembler von Argonaut

Aus London kommt ein integrierter Editor/Assembler, der mindestens ein MByte Arbeitsspeicher benötigt. ArgAsm, so die Bezeichnung, soll laut Hersteller einhundertmal so schnell sein wie der Commodore-Assembler von Metacomco und fünf- bis zehnmal so schnell wie seine zeitgenössischen Mitstreiter. Der integrierte Editor beinhaltet die besten Features des Devpac-Assemblers und das BRIEF, das auf MS-DOS-Rechnern verbreitet ist. Durch die Benutzung von Blitter-Routinen ist ein schnelles Text-Scrolling möglich. Die Kompatibilität zu anderen Assemblern wie Devpac oder Assem ist gewährleistet. Er wird auf zwei Disketten mit einem umfangreichen Handbuch (zur Zeit in englischer Sprache) ausgeliefert.

Bezugsquelle:
Argonaut Software Ltd.
7 Millway, Mill Hill
London NW7 3QR
Tel.: (0044) 906 3744
Fax: (0044) 959 5052

Designerträume werden wahr

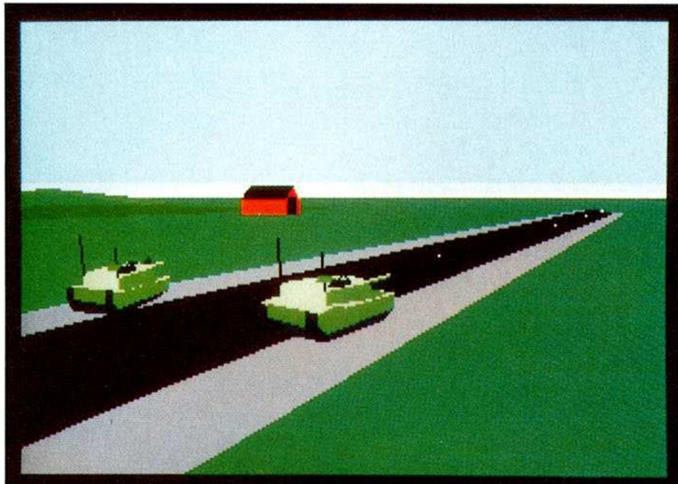
Der X-CAD Designer ist ein voll ausgestattetes 2D-Design- und Konstruktionsprogramm, das mit knapp 300 DM im unteren Preisniveau liegt, was jedoch nicht bedeutet, daß es auch im unteren Leistungsbereich liegt. X-CAD Designer ist dank der Benutzerführung durch Pulldown-Menüs einfach zu handhaben. Das Besondere an diesem Programm ist die Lese- bzw. Schreiboption von AutoCAD DFX sowie seine Verarbeitungsgeschwindigkeit. Die Eingabe-Charakteristiken können unter anderem auch in kartesischer Form vorgenommen werden.

Der X-CAD Designer ist auf allen Amiga-Modellen lauffähig, die über ein MByte Arbeitsspeicher verfügen. Hier noch ein paar Features von X-CAD Designer:

- Auto-Dimensioning mit Parametern konstruierbar
- Textmodifizierung (metrische und englische Real-Word-Einheiten)
- Konstruktionshilfen für Linien und Bögen (beispielsweise für Tangentenerstellung)
- Unterstützung von Pen-Plottern und Matrix-Druckern

Bezugsquelle:
Hazlitt Mews
Hazlitt Road
London W14 0JZ
Tel.: (01) 603 3313
Fax: (01) 602 2726

M1 Tank Platoon



Screenshot: M1 Tank Platoon

Die Simulationsprofis von Microprose haben wieder zugeschlagen. Nach U-Booten, Bombern und Düsenjets dürfen Heimcomputerbesitzer nun auch ein Tank Platoon kommandieren. Eine Platoon-Formation besteht aus fünf Panzern und sechzehn Soldaten, die zu einer Kampfeinheit verschmolzen werden müssen.

Der Realismus, mit dem Microprose-Simulationen entwickelt werden, feiert hier wieder fröhliche Urstände. Ein 200seitiges Handbuch erläutert nicht nur die Spielmechanik, sondern vermittelt auch sehr viel Praxiswissen über die Bedienung eines Panzers im Kampfeinsatz. Neben verschiedenen Terrains, die im Spielablauf auch ausgenutzt werden können (hinter Hügeln kann man sich verstecken), erwarten den Spieler auch verschiedene Wetterverhältnisse und viele High-Tech-Waffensysteme. M1 Tank Platoon wird voraussichtlich im Frühjahr 1990 für den Amiga fertiggestellt werden.

Computerspiel für flinke Zeichner

Pictionary nennt sich ein neues Gruppenspielkonzept, bei dem es darum geht, einen Begriff zeichnerisch auf dem Bildschirm darzustellen und von den Mitspielern erraten zu lassen. Die Begriffe werden mit einem integrierten Malprogramm gezeichnet, wobei das Spielgeschehen vom Amiga verwaltet wird. Das Computerspiel von der Firma Domark kostet zirka 90 DM und ist im Fachhandel erhältlich.



Screenshot: Pictionary

Drachenatem

Für das 1990er Frühjahr kündigt Palace Software ein neues Spiel namens Dragon Breath an. Der Spieler wird ins magische Land von Anrea versetzt und muß dort die Geheimnisse des alten Zwergenberges ergründen, von dem es heißt, hier sei ein Elixier versteckt, das Unsterblichkeit verheißt. Bis zu drei Spieler können an der Suche teilnehmen, wobei jeder Spieler auch vom Computer verkörpert werden kann. So sind Ein Spieler-, Zwei-Spieler- oder Drei-Spieler-Partien möglich. Das eigentliche Spielgeschehen vereinigt Rollenspielelemente genauso wie Strategie und Handelskomponenten, und nur wer alle drei meistert, hat eine Chance, das Spiel zu gewinnen.



Screenshot: Dragon Breath

Pionierleistung in der Informationstechnologie

Am 25.11.1989 konnten die Zuschauer der Fernsehsendung "Highscore" im dritten Programm des WDR eine Weltneuheit miterleben. Erstmalig wurde Software mit dem neuen Videodat-System zusammmen mit dem Fernsehbild gesendet. Wer über einen Fernseher, einen Videodat-Decoder und einen Computer (in diesem Fall ein Amiga) mit beliebiger DFÜ-Software verfügt, konnte sich eine Demo-Version des Fußballspieles "Bodo Illgners Super Soccer" in den heimischen Computer einladen, abspeichern und anschließend probespielen.

Bei der neuen Technik Videodat handelt es sich um ein Verfahren, mit dem Computerdaten in Bilddaten umgewandelt werden und in der sogenannten Austastlücke des Fernsehbildes gesendet werden. Pro Stunde können auf diese Weise 2,9 MByte Daten gesendet werden, die über den Videodat-Decoder wieder in digitale Daten zurückgewandelt werden.

Der Versuch, nicht nur Textdateien, sondern vollständige Computerprogramme auf diese Art zu übermitteln, ist mit dieser Fußball-Simulation auf Anhieb gelungen. Die Datensicherheit dieser Übertragungsmethode kann somit als sehr hoch bezeichnet werden.

Soundmonitor von Demonware

TFMX nennt sich ein neuer Soundmonitor von Chris Hülsbeck, der für die musikalische Gestaltung bei diversen kommerziellen Spielen (R-Type, Rock'n'Roll X-Out) verantwortlich ist. Über die Fähigkeiten dieses Musikprogramms für den Heimanwender, werden wir in einer unserer nächsten Ausgaben berichten.



Covergrafik David Wolf – Secret Agent



Hier wird Angela Clement alias Kelly ONeal für die Titelsequenz fotografiert

Jerry Luttrell

David Wolf Secret Agent

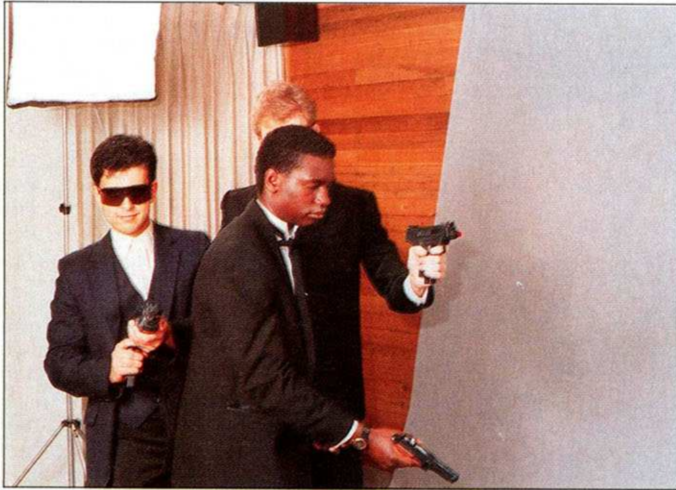
Hinter den Kulissen von Dynamix

Spiele, die auf irgendwelchen erfolgreichen Kinofilmen basieren, gibt es zuhauf. Ein Spiel wie ein Film gibt es noch nicht. Aber das wird sich ändern. Wer sich die Mühe macht, das renommierte US-Softwarehaus Dynamix zu besuchen, der kann vielleicht einen Blick auf die laufende Produktion des neuen Spiels David Wolf – Secret Agent werfen. Und dabei drängt sich automatisch die Frage auf: Wird hier ein Film gedreht oder ein Computerspiel gemacht?

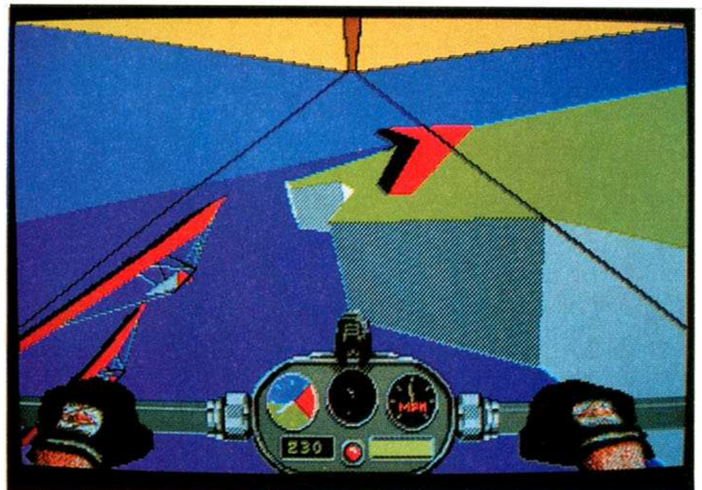
Als Anfang 1989 die Produktion von "David Wolf: Secret Agent" begann, wußten wir, daß unsere Vorstellungen von diesem Spiel anders waren als alles, was es bisher gab, und daß dieses "neue Konzept" einen völlig neuen Produktionsprozeß erfordern würde. Dieser Prozeß würde auch eine Anzahl neuer Technologien beinhalten. Wir wollten 3D-Technologie mit Bitmap-Animationen verknüpfen. Wir wollten "animated story sequences" in 3D, und vor allen Dingen wollten wir die Grafiken digitalisieren, um dem Spiel ein realistisches Aussehen und Feeling zu geben. Zusammen mit der neuen grafischen Technik kam die Überlegung, daß das Spiel ursprünglich im 16-Farben-IBM-EGA-Mode hergestellt wurde und dann die 256-Farb-VGA- und die Commodore Amiga-Versionen folgten. Um die Grafiken der VGA-Karte und des Amiga voll ausnutzen zu können, war frühzeitiges Planen nötig, um sicherzustellen, daß die Grafik für jede Version die Möglichkeiten des einzelnen Systems voll ausschöpft.

Das System zur Erzeugung und später zur Umwandlung der digitalisierten Grafiken war fürchterlich unausgereift.

Ein Projekt, das digitalisierte Grafiken in dem Umfang benutzte, wie wir es vorhatten,



Auch die bösen Buben des Spiels werden durch richtige Schauspieler dargestellt und später digitalisiert



Flucht mit einem Drachenflieger, eine der 3D-Sequenzen des Spiels

hatte es noch nie gegeben. Ohne jegliche Ahnung, wie wir anfangen sollten, wandten wir uns an Zeichentrickfilmhersteller. Seit Jahren waren die Prozesse in der Filmproduktion optimiert worden, um die Kosten zu senken und um hohen Qualitätsansprüchen zu genügen. Also fingen wir so an, wie man auch im Filmgeschäft beginnt: mit einem Drehbuch und einem Storyboard (skizzierter Ablaufplan). Mit dem Drehbuch kamen wir ganz gut voran, mit der Einschränkung, daß es eine Vielzahl von möglichen Enden und Handlungspfaden hatte. Anders als Drehbücher für Zeichentrickfilme, die von Anfang bis Ende eine klare Linie verfolgen, mußte unsere Geschichte sich verändern, je nachdem, was der Spieler tat. Das bedeutete, daß jede Szene mehrere Enden haben würde, wobei jedes Ende in einer anderen Richtung auf das Ende der Geschichte hinauslaufen würde. Es war kompliziert, alle verschiedenen Handlungspfade richtig zu verfolgen, aber das war noch gar nichts, verglichen mit dem Aufwand, der für die Erstellung des Storyboards nötig war, als das Drehbuch fertig war.

Das Storyboard – Hier werden Szenen und Bewegungsabläufe festgelegt

Das Storyboard ist eine detaillierte, visuelle Beschreibung der künstlerischen Anforderungen, Bewegungen der Darsteller und Dialoge in jeder Szene. Zeichentrickfilmhersteller verwenden Storyboards, um eine Szene auf preiswerte Art zu gestalten und um zu sehen, ob alles so funktioniert, wie es geplant war. Wir benutzten das Storyboard für den gleichen Zweck. Es war äußerst wichtig, jede Szene nur ein einziges Mal zu drehen, weil wir die Schauspieler heranbringen, Außenaufnahmen machen und gleichzeitig Grafiken für drei Systeme (16-, 256-Farb-VGA und Amiga) erstellen mußten. Unser Budget erlaubte uns einfach keine größeren zusätzlichen Aufnahmen, und niemand konnte uns garantieren, daß wir die Schauspieler wieder bekommen würden, wenn eine Szene erst einmal abgedreht war. Der Vorgang der Erstellung des Storyboards stellte sich als unglaublich langwierig, aber sehr wichtig heraus. Außer dem vorzeitigen Gestalten von Szenen, wo Schauspieler auftraten, war es sehr nützlich, daß wir für die 3D-Animationsequenzen ein Storyboard machten. Indem wir dies für jene Sequenzen zusammen mit den digitalisierten Szenen taten, konnten wir das gesamte Spiel auf dem Papier sehen, bevor es tatsächlich produziert wurde. Als das Storyboard fertiggestellt und einzelne Szenen bearbeitet worden waren, begann die Produktion des Computergrafikspiels in zwei unterschiedlichen Lagern: zweidimensionale und dreidimensionale Grafiken. Die 2D-Abteilung war dafür verantwortlich, die Schauplätze, Personen und digitalisierten Bilder zum Leben zu erwecken, während das 3D-Team die zahlreichen 3D-Animationen zusammenbaute, die David Wolf erforderte.

Völlig einzigartig bei der Produktion von David Wolf war die Arbeit mit lebenden Schauspielern an realen Schauplätzen. Als am Anfang der Entwicklung die Entscheidung getroffen wurde, digitalisierte Grafiken zu benutzen, wußten wir, daß wir



Besprechung der nächsten Szene anhand des Storyboards. Hier sind alle Handlungsabläufe festgehalten



Greg Sheid, der den David Wolf darstellt, beim Make-up

mit Rollenbesetzungen und Bühnenbildern zu tun haben würden. Insgesamt wurden 18 Rollen verteilt, und wir brauchten fast vier Wochen, um die verschiedenen Sequenzen des Drehbuchs zu drehen. Weil David Wolf sowohl bewegungslose als auch "animated", also bewegte, digitalisierte Teile kombiniert, verwandten wir letztendlich stille und Videofotografien, um Bilder einzufangen. Für die Standbilder wurden die Schauspieler und Bühnenbilder mit einer 35-mm-Kamera fotografiert, und diese Farbstandbilder wurden dann in den Computer übertragen, wofür wir einen Farbscanner benutzten. Bei den Animationssequenzen war der Vorgang nicht so einfach. Wir entschlossen uns, eine Art elektronischen Rotoscopings (Animationssequenzen, die nach Realbildern gezeichnet wurden) zu benutzen. Die Live-Handlungen wurden auf Video aufgenommen und dann elektronisch auf dem Computer deckungsgleich überarbeitet. Das ist ein unglaublich zeitaufwendiger Vorgang, aber das Endergebnis ist alle Mühe wert. Wenn die Grafik vom Film auf den Computer übertragen wurde, beginnt ein anderes Team damit, die Farbwerte anzupassen und alles nochmals zu bearbeiten. Die Farbwertanpassung nimmt wirklich sehr viel Arbeit in Anspruch. Bei der Mehrzahl der Aufnahmen begannen wir mit einem Bild, das 256 Farben verwendete und im IBM-Format war. Die Farbwerte werden dann an 32 Farben beim Amiga und 16 Farben bei der IBM EGA-Version angepaßt. Ist diese Anpassung fertig, muß der Künstler nochmals alle Versionen durchgehen und die Bereiche überarbeiten, die während der Farbveränderungen verzerrt wurden. Bei den EGA-, VGA- und Amiga-Versionen mit über 400 Aufnahmen für David Wolf wurde der Arbeitsaufwand immens. Manchmal brauchten wir bis zu sieben Künstler, um die Arbeit fertigzustellen.

Während die 2D-Grafiken geschaffen wurden, hatte die 3D-Abteilung alle Hände voll zu tun, die Welten für die verschiedenen 3D-Sequenzen zu erstellen. Bei Dynamix haben wir ein ganzes System entwickelt, um dreidimensionale Welten und Formen zu erzeugen und zu sehen. Dieses



Action auf der Straße. In David Wolf – Secret Agent ist natürlich auch die obligatorische Verfolgungsjagd per Auto enthalten

System (3Space TM) erlaubt es Künstlern, farbige runde 3D-Formen in bis zu 256 Farben mit Lichtquellenschattierungen und einer Farbe im Farbverlauf (für entfernte Schattierungen) zu erzeugen und zu bewegen. Mit Hilfe von Grafikpapier und Gittermodellen werden zunächst alle Umrisse gestaltet.

2D- und 3D-Grafik werden in vielen Szenen miteinander kombiniert

Die Datenpunkte der Grafikpapierzeichnungen werden dann mit einem 3D-Editor elektronisch reproduziert. Wenn die Punkte in den Editor eingegeben sind, werden den verschiedenen Objekten Farben, Orientierungshinweise (damit der Computer weiß, welche Seite innen und welche außen ist) und andere Informationen wie Bewegungen oder Lichtquellenschattierung zugeteilt. Wenn alle Formen produziert sind, werden sie in den Szenen zusammengebracht. Bei der Szene, in der David Wolf die Klippen von Dover hinabsegelt, sind zahlreiche 3D-Formen zusammengebracht worden, um eine 3D-Welt zu erzeugen: die Klippen, das Wasser darunter, sieben Segelflieger und ein Boot. Wie bei allen 3D-Formen müssen den Objekten spezielle Befehle gegeben werden, damit sie richtig agieren. Die Segelflieger müssen wie Segelflieger reagieren, sich mit den Luftströmungen auf und nieder bewegen. All diese Zauberei wird von Programmierern gemacht, die die Formen aufbauen und ihnen sagen, wie sie aufeinander reagieren sollen. Nachdem die 3D-Leute die Formen erzeugt haben und die Programmierer diesen gesagt haben, was sie tun sollen, werden all diese Objekte zusammen in eine einzige 3D-Landschaft eingebaut – und eine 3D-Welt ist geboren.

Als die 3D- und 2D-Grafiken fertig waren, war es Zeit, die beiden Elemente miteinander zu kombinieren. David Wolf ist eines der ersten Computerspiele, in denen in diesem Ausmaß 2D- und 3D-Grafiken verbunden werden, und wir fanden, daß mit dieser Technik erstaunliche cinematographische Effekte erzielt werden können. David Wolf enthält viele Szenen, in denen sich ein 2D-(bitmap-)Bild durch eine 3D-Welt bewegt. Die Kombination dieser zwei Methoden verleiht den Computerspielen eine neue Realitätsnähe. Im folgenden ist das Beispiel des Aufbauvorgangs einer 2D-3D-Kombination erklärt.

In einer einzigen Szene steckt mehr Arbeit, als es den Anschein hat

In dieser Sequenz dringt Wolf in die Basis der bösen Männer, Viper, ein. Die Aufnahme ist im Konzept ziemlich einfach; sie zeigt Wolf von der Seite, wie er in eine Flugzeughalle läuft und den gestohlenen Stealth Fighter (neuer Flugzeugtyp; nicht über Radar zu entdecken) untersucht. So wurde die Szene produziert:

1. Für die gesamte Szene wurde ein Storyboard erstellt. Wir mußten sicher sein, daß die Schnitte zwischen den Szenen gut sind und daß die Kombination der 2D-Animationen und der 3D-Welt stimmt.

2. Der Schauspieler, der David Wolf spielt, wurde mit Video beim Laufen aufgenommen und dann elektronisch "gerotoskopt". Dazu griffen wir acht Aufnahmen aus dem Videofilm heraus. Nach dem Übertragen auf den Computer versahen wir die Bilder einzeln mit Farben.

3. Die 3D-Formen wurden erstellt, indem wir ein Modell gebaut, das dann auf Grafikpapier übertragen und in den 3D-Editor eingegeben haben.

4. Aus den 3D-Formen wird die 3D-Welt zusammengebastelt.

5. Die Animationssequenz wird in die 3D-Welt eingepaßt. Die Bewegung des laufenden David Wolf muß Bild für Bild eingepaßt werden, um die Illusion zu erzeugen, daß er sich durch die 3D-Welt bewegt.

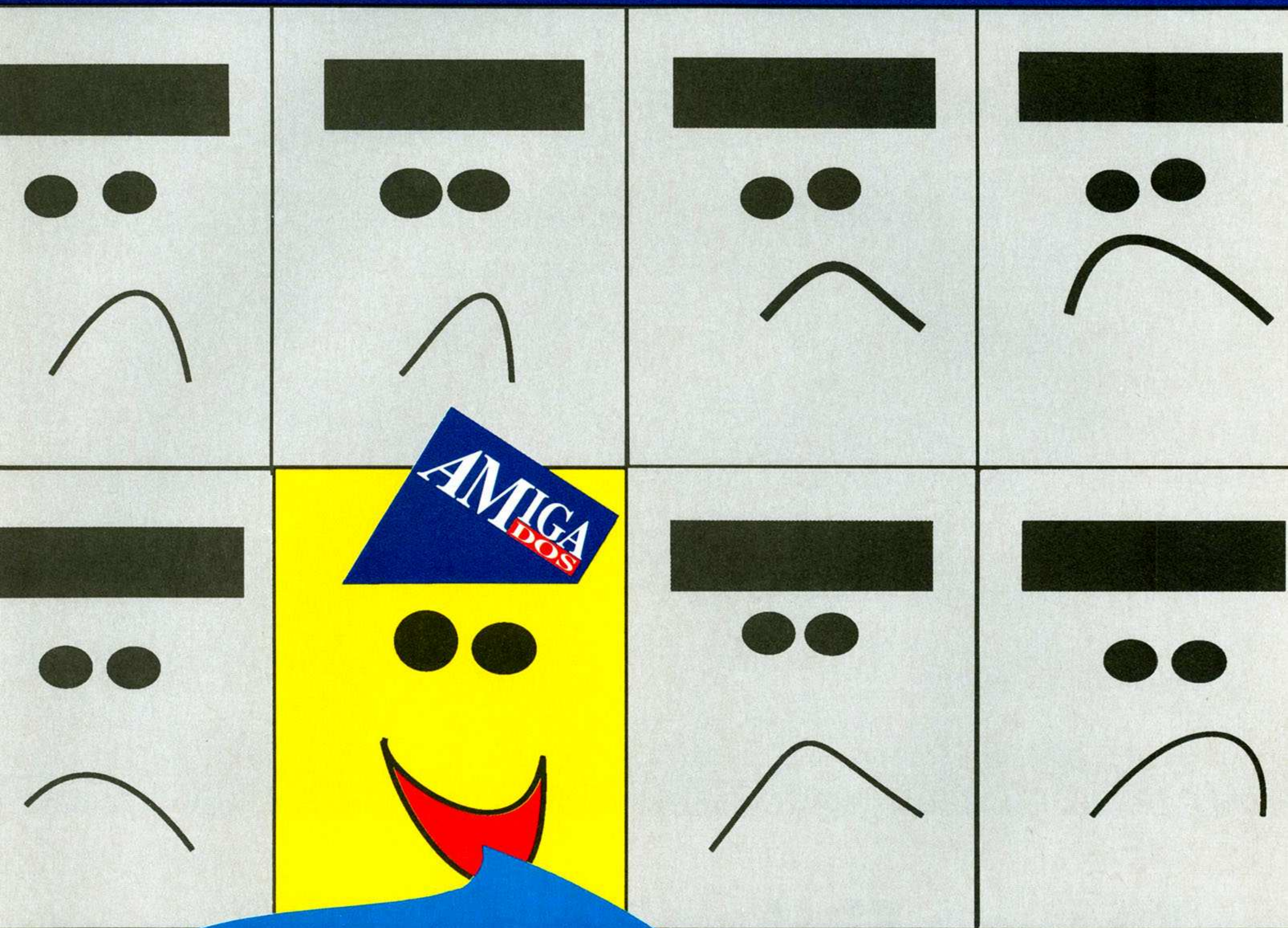
6. Wolf wird gezeigt, wie er das gestohlene Flugzeug untersucht. In dieser Sequenz fährt die Kamera um den Stealth Fighter herum. Diese 3D-Drehung ist ein Effekt, der in einer 2D-Landschaft fast unmöglich wäre.

Während David Wolf eine der innovativsten Techniken präsentiert, die je in einem Produkt der Computerunterhaltung verwendet wurden, ist manches davon mittlerweile schon wieder überholt worden von Projekten, die laufend von Dynamix entwickelt werden. Die Zukunft verspricht emotional ansprechendere Geschichten, mehr Interaktion, größere und bessere 3D-Welten und mehr Animationen von höherer Qualität. Mit CD-ROM, Bewegtbildern mit 24 Aufnahmen pro Sekunde und digitalisiertem Sound (keine Textboxen mehr) ist die Zukunft der Computerunterhaltung unbegrenzt.

Von Jerry Luttrell
Direktor of Marketing,
Dynamix, Inc.

(hs)

ABONNEMENT



INFORMATIONEN AUS ERSTER HAND

Ein Abonnement ist praktisch und bequem.

Widerrufsrecht
Jeder Abonnent hat das Recht, seine Bestellung innerhalb einer Woche beim DMV-Verlag, Postfach 250, 3440 Eschwege, schriftlich zu widerrufen. Die rechtzeitige Absendung des Widerrufschreibens genügt zur Fristwahrung.

AMIGA DOS
kostet im Abonnement:

Im Inland bzw. West-Berlin:
6 Ausg. = 35,– DM
12 Ausg. = 70,– DM

Im europäischen Ausland:
6 Ausg. = 50,– DM
12 Ausg. = 100,– DM

Im außereuropäischen Ausland:
6 Ausg. = 60,– DM
12 Ausg. = 120,– DM

Bitte benutzen Sie die Bestellkarte.

DMV-Verlag · Postfach 250 · 3440 Eschwege

DMV Daten- und Medienverlag



Die Videozauberer aus Kansas

Newtek stellt sich vor

Der Amiga ist ein Computer mit enormer Grafikpower, soviel steht schon einmal fest. Was passiert jedoch, wenn sich eine Gruppe routinierter Hard- und Softwareentwickler zusammensetzt? Gibt es noch weitreichendere Einsatzmöglichkeiten für den Amiga? Kann er mehr, als es zunächst den Anschein hat? Er kann, wie die Forschungsergebnisse von Newtek in eindrucksvoller Weise aufzeigen. Digi-View, das erste Projekt in der Zielrichtung Video-Amiga zeigte, was alles mit dem Amiga in Verbindung mit einer Kamera möglich ist. Doch war dies erst die Spitze des Eisbergs? Ja, denn mit dem Videotoaster liegt eine neue Generation der Videoverarbeitung vor. Bildumsetzung in Farbe und Echtzeit sowie die Möglichkeit, zeitgleich Effekte zu realisieren, war das Ziel, als die Arbeit am Videotoaster begann. AMIGA DOS hat daher für Sie die Firma Newtek in Kansas besucht und ist diesen Fragen auf den Grund gegangen.

Newtek hat seinen Hauptsitz in Topeka, einer 180.000-Seelen-Gemeinde inmitten der Kornkammer Amerikas.

In der dortigen Abgeschiedenheit entwickelt und produziert Newtek seine Hard- und Softwarepalette, die ausschließlich auf den Amiga zugeschnitten ist. Die Räumlichkeiten bestehen aus einem Bürogebäude sowie einer weiteren Forschungsstation, die von den Mitarbeitern liebevoll Alcatraz genannt wird. Nicht einmal alle Mitarbeiter von Newtek haben Kenntnis von dieser Einrichtung bzw. wissen genau, wo Alcatraz zu finden ist.

Videotoaster, was ist das?

Nur soviel sei gesagt: Ein Unbedarfter würde nie hinter diesen Mauern eine Soft-und Hardwareforschungsstation vermuten. Doch was ist nun ein sogenannter Videotoaster? Was steckt hinter diesem Kürzel? Zu diesem Zweck befragten wir Tim Jenison und Paul Montgomery.

Der Videotoaster ist eine Spezialhardware, die es ermöglichen soll, ein Videosignal in Echtzeit auf den Monitor zu bringen. Gleichzeitig besteht die Möglichkeit, dieses Videosignal zu verfremden und damit aufsehenerregende Effekte zu erzielen. Dabei sieht der Videotoaster recht unscheinbar aus. Er besteht aus einer einseitig bedruckten Platine, die im Videoslot des Amiga angeordnet ist. Ein Kabel verbindet die Karte mit der Videokamera, ein zweites verbindet den Toaster mit einem Oszilloskop, ein drittes schließlich stellt die Verbindung zum Videomonitor dar.

Der Toaster nimmt nun ein bestehendes Bild von der Videokamera auf. Auf dem Videomonitor kann man dann sehen, welche Möglichkeiten in dem Videotoaster stecken. Ob Zerbröselungseffekte, Seitenflips, Aufziehen des Bildes auf eine Kugel oder der bekannte Mosaikeffekt, der Videotoaster verfügt über zirka 20 verschiedene Möglichkeiten, ein Bild zu manipulieren bzw. zu verfremden. Viele der Effekte können unterschiedlich stark ausfallen. Der Mosaikeffekt mag da als Beispiel dienen. Der Effekt kann bewirken, das der Bildschirm in vier, acht oder sechzehn Teile aufgeteilt wird, und in jeweils einem dieser Teile wird das komplette Bild reproduziert. Auf diese Weise entsteht der Eindruck einer Bildwand, wie Sie ja aus den diversen Fernsehstudios bekannt ist.

Fast unbegrenzte Möglichkeiten

Aber noch viel mehr ist möglich. So gibt es einen Effekt, der besonders, wenn es um Personen geht, erstaunliche Ergebnisse zeigt: der Spiegeleffekt.

Man kann nämlich die Hälfte eines Bildes an einer senkrechten oder horizontalen Achse spiegeln, und auf diese Weise interessante Wirkungen erzielen. So entstehen im Handumdrehen völlig neue Gesichter, die an einen Besuch im Spiegelkabinett erinnern. Natürlich lassen sich die Effekte wirkungsvoll miteinander kombinieren. Verwendet man beispielweise gleichzeitig den Spiegeleffekt mit dem Mosaikeffekt, ergibt sich ein Bild, das von oben nach unten aufgeteilt ist. Die einzelnen Teile können nach rechts oder links gedreht werden. Es können aber auch Bilder in sich drehende Teile eingeblendet werden! Eine weitere Option zeitigt einen Linseneffekt, wie wir ihn von der Kamera vom Weitwinkel/ Fischauge her kennen. Das das Bild an jede beliebige Stelle des Bildschirmes verschoben werden kann, versteht sich von selbst. Aber auch ein Zoomeffekt ist möglich, der das Bild bis zur Pixelstruktur auflöst.

Oder wollen Sie ein Bild aus einem beliebigen Punkt irgendwo auf dem Bildschirm entstehen lassen? Der Verschiebungs- und Zoomeffekt macht's möglich.

Die Technik

Prinzipiell kann man also sagen, der Toaster ist eine Spezialhardware und erzeugt Sondereffekte, die prinzipiell durchaus mit denen der professionellen Fernsehstudios übereinstimmen. Die von der Kamera aufgenommenen Signale werden von der Hardware erst einmal gespeichert und können anschließend von der Toasterebene aus gezeigt werden. Im Speicher des Toasters gehen dann auch die zahlreichen Verfremdungen vor sich, die die Sondereffekte erzeugen.

Leider ist der Toaster momentan nur auf NTSC, das amerikanische Fernsehsystem, ausgelegt. Ein Teil des Toasters, der Digitizer muß nun mindestens 16 Halbbilder pro Sekunde digitalisieren und auf den Schirm bringen, wenn eine flüssige Bewegung entstehen soll. Alles, was langsamer ist, wird vom menschlichen Auge nicht mehr als flüssige Bewegung erkannt und verrät sich durch große Motivsprünge, die sich natürlich störend auswirken. Doch Newtek hat es geschafft, genügend Geschwindigkeit beim Digitalisieren in Echtzeit aufzuweisen. Im Video-Display-Buffer, indem das eingehende Bildsignal mit Hilfe der Software verfremdet wird, wird das Signal dann auch gleich in ein sendetaugliches RS-170A-Videosignal umgewandelt. Das so vorliegende Signal kann nun entweder gesendet, auf dem Monitor angezeigt oder auf Video gespeichert werden. Allerdings, bei aller Euphorie hat auch der Videotoaster seine Grenzen. So ist es beispielsweise nicht ohne weiteres möglich, das Bild eines Heimvideorecorders zu verarbeiten. Der Grund: der Videotoaster ist auf NTSC ausgelegt und nur schlecht in der Lage, ein PAL-Bild zu synchronisieren.

Aber auch die Ausgabe von Bildern weist Begrenzungen auf. So ist es nicht möglich, auf der gleichen Zeile ein HAM- oder Halfbrite-Bild mit einem Signal des Toasters zu verarbeiten. Soll dieser Vorgang trotzdem stattfinden, das heißt, Amiga-Grafiken und Toasterbilder sollen gleichzeitig vorhanden sein, dann hilft nur die Aufteilung des Bildschirms durch eine horizontale Trennlinie. Der eine Bereich zeigt dann die Amiga-Grafiken, der andere das Signal, das der Toaster aus dem eingehenden Videosignal gebildet hat.

Die Steuerung des Toasters

Wie jedoch werden so komplizierte und umfangreiche Effekte nun gesteuert? Am einfachsten geht dies natürlich mit Hilfe der beiliegenden Software, deren Icons lediglich angeklickt werden müssen, um den gewünschten Ef-



Bild 1: Hinter dieser unscheinbaren Fassade verbirgt sich der Hauptsitz von Newtek, wo Produktion, Verwaltung und Versand der Soft- & Hardware stattfinden

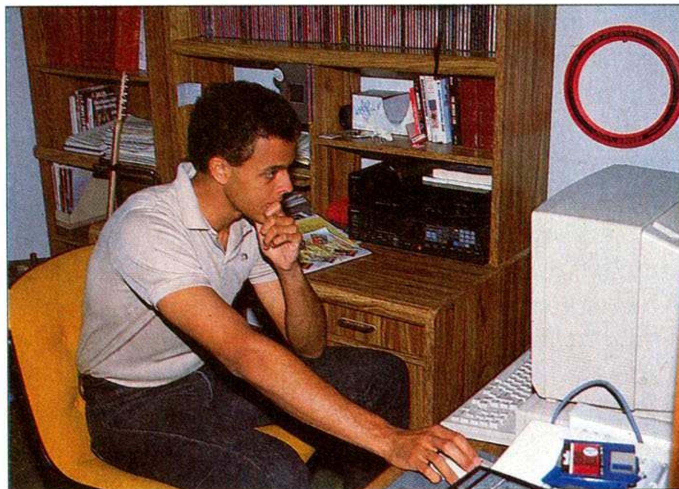


Bild 2: Robert Blackwell, einer der Programmierer von Newtek bei der Arbeit an einem neuen Projekt

fekt hervorzurufen. Hier kann auch auf einfachste Weise mittels Mausklick eine Kombination der vielen Effekte vorgenommen werden. Es stehen jedoch noch zwei weitere Möglichkeiten der Steuerung zur Verfügung. So kann man mit seinem Amiga Scriptanweisungen erstellen, die nacheinander die Effekte aufführen, die zum Einsatz kommen. Dieses "Programm" wird dann Schritt für Schritt abgearbeitet. Auf diese Weise läßt sich schon recht genau arbeiten, denn alle Parameter wie Art des Effektes, Dauer des Effektes oder dessen Geschwindigkeit können präzise vorgegeben werden. Auf diese Weise wird der Amiga zum Regisseur. Doch damit nicht genug, auf der dritten Ebene gibt es noch ATL (A Toaster Language). Sie stellt denen weitreichendste Möglichkeiten zur Verfügung, die Effekte gerne selbst programmieren wollen. Natürlich gehört schon ein gerüttelt Maß an Kenntnissen und Erfahrungen dazu, um auf diesem Wege die Möglichkeiten des Toasters zu erweitern. Immerhin beschäftigt sich Newtek ernsthaft mit dem Gedanken, eine Unterstützung für Fremdentwickler anzubieten. Die Programmiersprache für die Effekte ist übrigens Assembler. Auf diese Weise wäre es möglich, eine gute Kontrolle über den Videotoaster zu bekommen, ein anstrengender Weg, der natürlich auch entsprechende Programmierfähigkeiten voraussetzt.

Zukunftsmusik

Im Zuge der Arbeiten am Videotoaster, die nun bereits zehn Mannjahre in Anspruch genommen hat, sind natürlich ständig neue Ideen zusammengekommen. So plant man beispielsweise, einen Videotitler mit in den Toaster zu integrieren.

Die Vorteile liegen klar auf der Hand: Man kann seine Werke gleich betiteln und braucht nicht auf einen gesonderten Arbeitsgang zurückgreifen.

Die Wiedergabe des Videotitlers soll direkt in den Frame-Buffer des Videotoasters geleitet werden und auf diese Weise zusammen mit dem Hauptsignal zur Verfügung stehen. Doch der Videotoaster ist noch um einiges ausbaufähig.

Als eines der ersten Zubehörteile soll ein Time-Base-Corrector auf den Markt gelangen, der es dann erlaubt, auch einen handelsüblichen Heimvideorecorder an den Toaster anzuschließen und als Signalquelle zu nutzen.

Auch geplant sind RAM-Erweiterungen für den Toaster, um längere Videosequenzen und Animationen ohne Probleme handhaben zu können. Allerdings, die RAM-Preise sind auch hier ein Indikator, der über Erfolg und Ernüchterung entscheidet. Denn etwas RAM muß schon her, um den Videotoaster funktionsfähig zu machen. Bei den aktuellen RAM- Preisen, die ja ständig starken Schwankungen unterliegen, bedeutet dies, das der Videotoaster schon ein gewisses Loch in die Haushaltskasse reißen wird. Unter den gegenwärtigen Voraussetzungen und mit den aktuellen RAM-Preisen gerechnet, wird der Videotoaster ungefähr 650 US$ kosten. Auf den ersten Blick ein stolzer Preis, der jedoch in die Nähe eines Trinkgeldes rückt, wenn man Preise von vergleichbaren professionellen Produkten betrachtet. Sobald jedoch die RAM-Preise etwas fallen, wird der Videotoaster in Produktion gehen (zur Zeit wird aufgrund der RAM-Preise immer weiter verbessert). Ein Sinken der RAM-Preise hätte natürlich auch eine Verbilligung des Toasters zur Folge, was allen Endverbrauchern zugute kommt.



Bild 3: Ein typisches Büro bei Newtek – Einen PC wird man hier nur selten finden. Alle anfallenden Arbeiten werden mit Hilfe des Amiga oder des Macintosh erledigt

Add Ones

Gearbeitet wird auch noch an Möglichkeiten, um weitere Newtek-Produkte zusammen mit dem Toaster betreiben zu können. So ist beispielsweise ein Zeichenprogramm in Arbeit, das die volle Farbpalette des NTSC-Systems ausnutzen soll. Dieses Malprogramm soll möglichst auch noch Aufnahme im Toaster finden. Die Möglichkeiten, die dieses nach sich ziehen würde, wären schlicht riesig.

Im Gespräch ist weiterhin ein 3D-Grafikprogramm, mit dem man direkt den Frame-Buffer des Videotoasters ansprechen kann. Dies alles ist natürlich auch eine Frage des RAMs, und hier ist der Punkt gekommen, wo man eben nur auf eine Preissenkung warten kann.

Es wäre natürlich viel komfortabler, wenn alle geplanten Erweiterungen und Extras bereits implementiert wären, umständliche Erweiterungen oder weitere Boxen auf dem Computerschreibtisch würden entfallen.

Nun ist es so, daß von dem Videotoaster bereits Prototypen existieren, die klaglos und mit sehr guten Ergebnissen Ihre Arbeit verrichten. Da aber durch die RAM-Preise der Preis des Toasters unattraktiv hoch wäre (zirka 1300 Mark sind ja schließlich kein Pappenstiel) hat sich Newtek entschlossen, zunächst noch mit der Auflage einer Serie zu warten. Denn schon bei der Einführung von Digi View wurde die Firmenidee deutlich: dem Heimanwender professionelles Werkzeug zu einem geringen Preis in die Hand zu geben, mit dem dieser allerdings auch etwas anzufangen weiß. Da wie gesagt zehn Mannjahre notwendig waren, um den Toaster bis zu diesem Punkt zu entwickeln, braucht sich Newtek kaum Sorgen um Mitbewerber zu machen. So leicht dürfte es nicht sein, diesen Technologievorsprung einzuholen.

Demnächst...

Wir würden an dieser Stelle sicher gerne mehr über den Videotoaster und seine Möglichkeiten bringen. Er stellt eine innovative und für den Heimanwender sehr attraktive Möglichkeit der Vernetzung seiner Unterhaltungselektronik dar. Sobald der Videotoaster in Serie gefertigt wird und auf den Markt gelangt, werden wir ihn noch einmal in Augenschein nehmen. Die Vorführung des Toasters war zwar mehr als interessant, im Sinne einer möglichst präzisen Berichterstattung wollen wir an dieser Stelle nicht mehr vorwegnehmen.

Ein ausführlicher Test unter Laborbedingungen und unter Zuhilfenahme von Meßapparaturen wird zeigen, ob die Serienproduktion des Toasters die Effekte zeigt, die für den Prototyp keine Schwierigkeiten darstellen. Ist des der Fall, wird der Amiga sicherlich auch in professionellen Anwenderkreisen noch mehr Zuspruch finden. Denn wer würde sich schon für ein professionelles, zehnmal teureres Gerät entscheiden, wenn der Amiga das gleiche zu leisten in der Lage ist. Angesichts des Preises ist der Toaster auch fürdie meisten wirklich Interessierten erschwinglich.

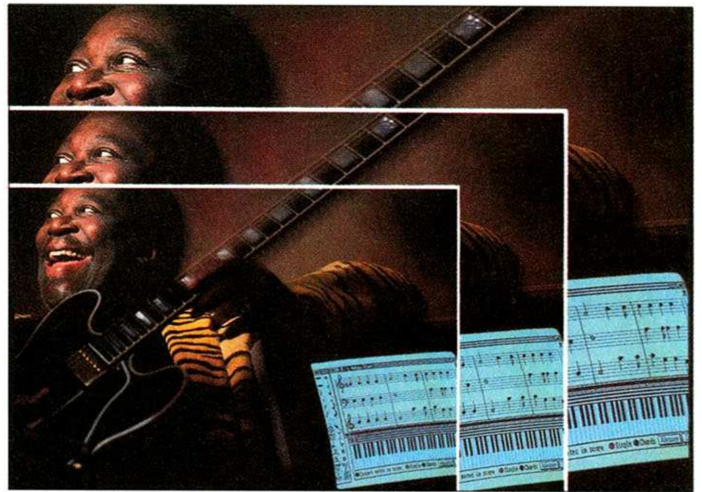
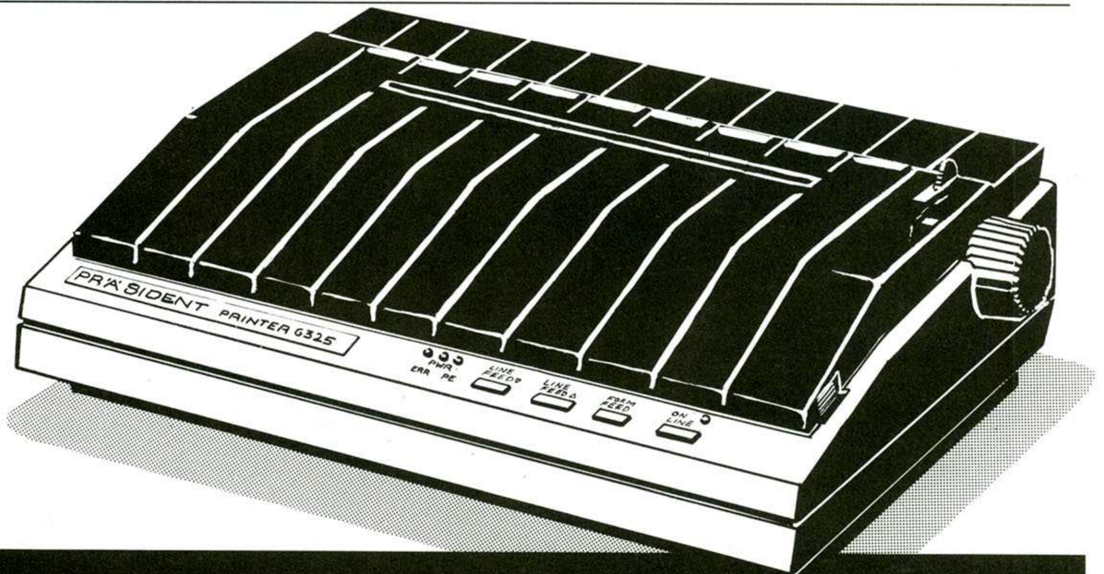


Bild 4: Bild im Bild – dies ist nur eines der Features, die der Videotoaster zu bieten hat

Über den Erfolg einer derartigen Hard- und Software lassen sich natürlich unter den gegenwärtigen Umständen keine Prognosen machen. Auf jeden Fall sollten die Anwender den Videotoaster aufmerksam im Auge behalten. Er könnte das Arbeiten mit dem Amiga leicht revolutionieren. AMIGA DOS wird für Sie die weitere Entwicklung des Videotoasters im Auge behalten und Sie informieren, sobald neue Daten über Preis, Leistungsumfang und Vertrieb vorliegen.

(mm)



Präsident Printer 6325

compatible zu fast allen Computern

Zeichen- und Befehlssätze:

Epson: Commodore/Centronics*
V24 RS232C = 100 %
Commodore compatible
Commodore 64 und 128 Zs und Bs
Amiga* Zs, IBM Zs und Bs I + II
Schneider* Zs und Bs
ATARI* ST Zs und XE/XL Zs und Bs
9 internationale Zeichensätze

Technische Daten:

- 9 Nadeln, 100 Z/Sek.
- Grafik Punktdichte/Zeile 480 min., 1920 max.
- bidirektional, druckwegoptimiert
- Druckarten: Normal, doppelt, breit, komprimiert, Sperrschrift, Exponenten/Indices, automatisches Unterstreichen, Near-Letter-Quality
- Verstellbare Stachelradwalze Einzugsschacht für Einzelblatt, Staubschutzhaube incl.

unverb. Preisempfehlung incl. Interface (wahlweise Centronics*, Commodore*, V.24/RS232C oder Atari* XE, XL)

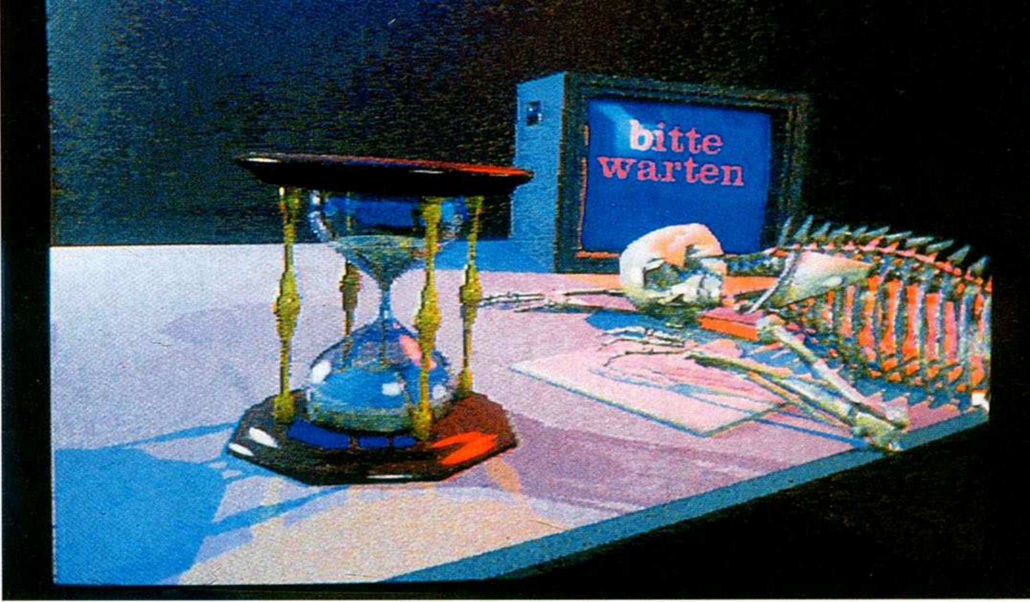
DM 349.–

NEU: Buffererweiterung bis 32 KB mit down-load-Funktion

Computerwechsel:
Interface tauschen, schon ist der Präsident Printer 6325 angepaßt.

Beratung: Grubert Service- u. Software-Center GmbH & Co. KG
8110 Waltersberg, Telefon 08847/6664

*Atari, Centronics, Commodore, Amiga, Epson, IBM, Schneider sind eingetragene Warenzeichen der Atari Corp., Centronics Data Computer Corp., Commodore Corp., IBM Corp., Schneider GmbH.



Ralph Conway

ANIMATION & GRAFIK

Vor rund eineinhalb Jahren habe ich den Begriff Desktop-Video (DTV) das erste Mal gehört. Was mir zunächst nur als eine interessante Variante zum Schlagwort Desktop-Publishing (DTP) erschien, erwies sich bei näherer Betrachtung als eigenständiger Anwendungsbereich.
DTV bezeichnet die Übernahme diverser Aufgabenstellungen aus dem Bereich der Videoproduktion durch den Computer und somit deren Verlagerung vom Studio zum Schreibtisch.

Dies bedeutet nicht etwa das das Studio seine Existenzberechtigung verliert, sondern vielmehr das Arbeiten, welche mit dem Computer genausogut oder besser realisiert werden können, als mit der reinen Studioausstattung eben auch mit dem Computer realisiert werden.

Der Computer wird als sinnvolles Werkzeug genutzt und übernimmt gerade auch im Studio alte und ebenso neue Aufgaben, die sich bisher nicht oder nur mühevoll und kostenintensiv realisieren ließen.

Desktop-Video auf dem Amiga?

Die Frage, warum sich für viele Aufgaben im Bereich der Videoanwendungen gerade der Commodore Amiga besonders eignet, ist sehr leicht zu beantworten.

Er liefert nicht, wie oft fälschlich behauptet wird, ein Videosignal; wie alle Computer generiert er ein RGB-Signal, welches mit dem benötigten Videosignal (bei uns FBAS-Pal) nicht kompatibel ist.

Aber im Gegensatz zu allen anderen Computern ermöglicht er serienmäßig eine Auflösung von bis zu 704 x 580 Bildpunkten. Diese Auflösung kommt dem eines Videobildes sehr nahe und ermöglicht eine einfache Übernahme der Bildinformation vom Rechner zum Videorecorder. Das Signal muß nur zu einem Videosignal gewandelt werden.

Aufgrund der vergleichbaren Auflösung besitzt das gewandelte Bild keine schwarzen Ränder wie bei anderen Grafikformaten (EGA, CGA, VGA).

Hinzu kommt neben der überragenden Grafikfähigkeit (bis 4096 Farben ohne Zusatzkarten) eine Bildwiederholfrequenz von 50 Hz und ein Bildaufbau im Interlace-Modus (mit Zeilensprung), also die Darstellung von 50 Halbbildern pro Sekunde. Dies entspricht technisch den bei der Pal-Norm gegebenen Fakten.

Das Problem des Signalwandelns ist somit auf ein techniches Minimum reduziert. Nämlich der Wandlung des RGB-Signals in ein FBAS-Palsignal.

Aufgrund dieser technischen Auslegung ist es außerdem recht einfach möglich, das RGB-Signal des Computers mit einem Videosignal quasi gleichzuschalten und beide Informationsquellen miteinander zu verknüpfen.

Ein Genlock-Interface erledigt diese Arbeit und ermöglicht es so, ein Videosignal z.B. mit Titeln, Grafiken oder digitalisierten Bildern, die aus dem Computer kommen, zu koppeln und beides gemeinsam auf Video aufzuzeichnen.

In diesem Artikel konkreter auf die diversen Möglichkeiten einzugehen, würde den Rahmen sprengen. Wir werden in den kommenden Ausgaben gezielt auf die diversen Einsatzmöglichkeiten ihres Computers (?) im Bereich Video zurückkommen.

Von der Titelgeneration über 2-D-Effektgeneratoren zur 2-D-Animation bis hin zur dreidimensionalen Computergrafik und Animation lassen sich die kreativen Tätigkeiten in der Gestaltung von Videofilmen ebenso realisieren wie die konzeptionellen Aufgaben (z.B. Storyboards, Kostenkalkulation) und die notwendigen Büroarbeiten (Buchhaltung, Personalien, Fakturierung).



Bild 1: Die Grafikmöglichkeiten des Amiga kommen aber besser bei Raytracing-Grafiken heraus...

Mindestvoraussetzung für die direkten Videoanwendungen sind ein RGB-Pal-Decoder oder ein Genlock-Interface, ein Megabyte RAM und Festplatte.
Je komplexer und rechenintensiver die Anwendungen sind, desto schneller stößt man an die Grenze der Leistungsfähigkeit dieser Grundausstattung.
Um eine komplexe dreidimensionale Animation im Ray-Tracing zu realisieren, bei welcher jeder einzelne Bildpunkt durch rechen- und somit zeitintensive Strahlenverfolgungsalgorithmen bestimmt wird und ein Bild auf einem "normalen" Amiga schon einmal 25 oder gar 40 Stunden Rechenzeit benötigt, empfiehlt sich von selbst der Einsatz eines Turboboards mit 68020 oder 68030 Prozessor und entsprechendem Co-Prozessor.
Je nach Taktfrequenz und Anpassung der Software und Ausstattung mit schnellem 32-Bit-RAM reduziert sich die Rechenzeit dann auf z.B. eineinhalb Stunden.

Die Entwicklung des letzten Jahres zeigt klar die Richtung auf, in welche sich der Bereich Desktop-Video bewegt. Der mittlerweile sehr professionellen Software folgt nun auch Hardware, die immer stärker professionellen Ansprüchen gerecht wird und neue Standards setzt.

Wohin weist die Zukunft?

Vom optischen Speichermedium, über Echtzeitdigitizer und Framebuffer, welche die Darstellung von 65000 und mehr Farben ermöglichen, bis hin zu Contollern für Einzelbildrecorder und Bandmaschinen, gelangt der Amiga immer weiter in professionelle Spähren in der Videoanwendung.
Der Commodore Amiga ermöglicht so auch dem ambitionierten semiprofessionellem Anwender den Einstieg in ein neues Medium und ist vom Startwerkzeug bis zum Profiinstrument in der Spezialanwendung aufwärtskompatibel.

(mm)

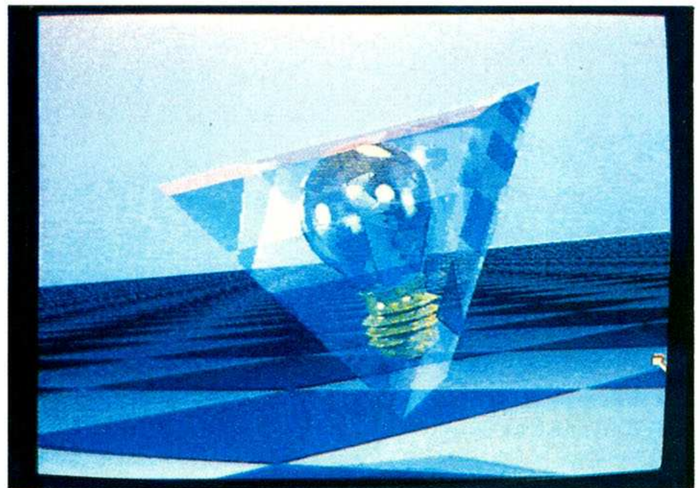


Bild 2: ... wie man in diesen beiden Beispielen sehen kann

Zum Autor:

Ralph Conway, geb. 02.05.1961 in Köln Mühlheim Diplomdesigner Kommunikations-Design FH. Seit 1984 freiberuflich tätig, ab 1987 mit Schwerpunkt Computergrafik und seit Juli 1989 als Anwendungsspezialist DTV bei der Commodore GmbH für die Anwendungsbereiche Grafik, Video und Animation zuständig.



Spitzen- Zubehör für AMIGA-PC's

128K RAM-Karte für XT-Karte: 248,00 DM
Kurze Karte mit 128K Ram. Somit haben Sie 640K DosRam und können z.B. Ventura, F&A und D-Base-IV anwenden !!!

XT-Multifunktions Karte 299,00 DM
Multi-IO-Karte für XT-Karte mit Ser., Par., Uhr & 128K Ram

EGA-Karte 640 * 480 198,– ~~280,00~~ DM
*Modi: EGA 640*350 / 640*480 / HGC / CGA - 256K Ram*

Dual-Video-Karte 138,00DM
Modi: CGA/HGC/MDA 100% AMIGA-XT kompatibel

Genius GM-6000 Maus (hochauflösend): 99,00 DM

30 MB XT-File-Card / 60ms 768,00 DM
50 MB XT-File-Card / 40ms 948,00 DM
Die File-Cards können für AMIGA & XT partitioniert werden.
Lieferung incl. ausführlicher Soft- & Hardware Installations-Anleitung

Fordern Sie unsere kostenlose Preisliste an

SKy-Ware, Postfach 1331, 5308 Rheinbach
Telefon: 02253/2667

Teletone

Telefax
von Panasonic, Mitsubishi, Schneider u.a.
48 Mon.
mtl. ab 49,-

Anrufbeantworter
50 verschiedene Modelle von Panasonic, Tiptel, Uher, Zettler u.a.
ab 198,-

Cityruf, Eurosignal, Telefonanlagen, sämtl. Zubehör. Katalog anfordern!

Nordring 98 · 4630 Bochum · Tel. 02 34/1 39 50 · Fax 1 49 31

WB·PD

AMIGA MS-DOS ATARI AMIGA MS-DOS ATARI

Auf 3,5" und 5 1/4"-Marken- und Qualitätsdisks liefern wir alle gängigen Serien!
ACS - AMIGA - AUGE 4000 - Amok - Amuse - Amsel - Amicus - Amiga Faszination - Amateurradio - BCS - Barrakuda - BPD - CSM - Cactus - Casa mi Amiga - Entertainment - Erotik - ES - Franz - Faug - Fred Fish - Get it - Kickstart - Kiss - MSS - Niclas - Public Project - Panorama - Poseidon - Pfalz - RMS - RPD - RW - Ruhr - Sale - Slideshow - Slipped Disk - SACC - Taifun - Taurus - Tornado - TBAG - Tail - U Kaug - WB PD Serie - Star Track - Return to Earth

Unser Angebot wird ständig erweitert!

PD-Preise inkl. Disk mit Qualitätsgarantie!
Mit doppeltem VERIFY kopiert! 3,5"-Disk. m. Etikett!

AMIGA	3,5"	5 1/4"			
1- 10	3,50	2.00	21- 50	2,90	1,40
11- 20	3,10	1,60	51-100	2,70	1,30
			ab 101	2,60	1,20

Preise für WB-PD-Clubmitglieder

	3,5"	5 1/4"
1-10 Stück	DM 2,50	DM 1,10
11-50 Stück	DM 2,20	DM 1,00
ab 51 Stück	DM 2,10	DM 0,90

Das goldene PD-Buch (Band IV) kompl. mit Disketten DM 110,--

Pakete für Einsteiger und Anwender (jeweils 10 Disketten)

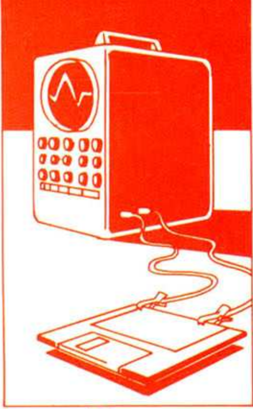
1. Einsteiger I: Tips und Hilfen, Spiele und Anwendungen für den Anfang
2. Einsteiger II: ... man braucht es eben ...
3. Spiele I: über 30 Spiele
4. Spiele II: ... 26 Superspiele, in Deutsch geschrieben oder dokumentiert
5. Sound: ... da geht die Post ab, Sounds zum Abspielen und Verändern (mit Player)
6. Grafik: ... mit Ray Tracing, DBW Render, Animation, Malprogramm
7. Modula II: Kurs in Deutsch und viel Zubehör

Jedes Einzelpaket nur DM 38,00
3 Pakete nach Wahl zus. nur DM 100,00

Superspiele III
10 Disketten - 3,5" DM 49,00

Versand nur gegen Vorkasse oder Nachnahme! Vorkasse: DM 5,-, Nachnahme: DM 8,-
Ausland: Nur Vorkasse DM 12,-
6 Katalog-Disketten DM 16,- (gegen Einsendung eines Schecks oder Briefmarken).

AMIGA PUBLIC DOMAIN
Wolfgang Bittner · Keltenstr. 15
6700 Ludwigshafen/Rhein 25
Telefon 06 21 / 67 49 64
Rund um die Uhr telef. Bestellannahme:
0 62 35 / 10 70



Mit Text und Bild

Das neue UBM-Text liest Grafiken ein

Bei allen Fähigkeiten des Amiga beachtet man eine besondere kaum: Er ist ein Computer, der auch zur Texterfassung herangezogen werden kann, wenn ein Programm dieser Art für ihn vorliegt. Die Firma UBM-Drecker GmbH beweist dies wieder, indem Sie eine neue Version ihres bekannten UBM-Text herausgibt. Wir haben uns das Programm für Sie angesehen.

Leider wird man als Besitzer eines Amiga viel zu oft von Kollegen, die einen PC, AT oder noch größeren MS-DOS-Computer ihr eigen nennen, mitleidig belächelt, wenn es um das Thema Texte und ihre computermäßige Verarbeitung geht. Ganz schön zu Unrecht, denn der Amiga hat eine Fähigkeit, die bei PC-Textverarbeitungen nur bei teuren Programmen zu finden ist: Er kann Bilder des IFF-Formates zu jeder Zeit einlesen. Und diesen Umstand macht sich auch das neue UBM-Text zunutze.

UBM-Text läßt individuelle Einstellungen beim Programmstart zu

UBM-Text V2.3 benötigt zur Arbeit einen Amiga mit mindestens 512 kByte, allerdings sind für alle Funktionen mindestens 1 MByte erforderlich, da sonst Grafikeinbindung, Video-Modus-Umschaltung und andere speicheraufwendige Operationen nicht benutzt werden können. Das Programm arbeitet mit einem Laufwerk, aber auch hier sind mindestens zwei Floppies zu empfehlen. Wohl dem, der über eine Festplatte verfügt. Der Textspeicher richtet sich nach dem verfügbaren Hauptspeicher, er kann je nach Vorhandensein erweitert werden, Besitzer einer Speichererweiterung sind also auch hier wieder im Vorteil. Voreingestellter Wert des Textspeichers ist 10 kByte, er kann jedoch, wie schon gesagt, erweitert werden, dies geht allerdings dann vom Hauptspeicher ab.

Für den Drucker existiert unter UBM-Text eine eigene Steuerzeicheneinstellung, bei vorhandener Grafik muß jedoch auf die Voreinstellungen von Preferences zurückgegriffen werden. Hier sollte man wieder mit dem eigenen Drucker erst einmal Versuche starten, bevor man ernsthaft loslegt.

Die Einstellungen für Textspeicher, Video-Modus usw. werden direkt als Parameter dem Hauptprogramm beim Aufruf übergeben:

'UBM-TEXT -m:50000 -i

zum Beispiel stellt 50 kByte an Textspeicher zur Verfügung und aktiviert den Interlace-Modus.

Der erste Text sollte zum Üben der Funktionen von UBM-Text dienen

UBM-Text verfügt über sämtliche Funktionen, die eine gute Textverarbeitung aufweisen sollte: einzeln auswählbare Schriftarten, Format- und Block-Befehle, Abspeichern der Texte im UBM-Text-Modus oder als reiner ASCII-Text, sowie WYSIWYG-Darstellung, es erscheint also das auf dem Bildschirm, was hinterher auch auf dem Drucker zu sehen ist.

Neu bei UBM-Text ist die Möglichkeit, Grafik-Dateien in bestehende Texte einzulesen. Die Texte werden dabei mit der Grafik zusammen auf dem Bildschirm dargestellt.

Um den größtmöglichen Erfolg beim Mischen von Text und Grafik zu erzielen, existiert auf der UBM-Text-Diskette die Datei 'Graph-Cut'. Diese zeigt nach Aufruf die vorhandenen Laufwerke an, deren Dateien sich per Maus auswählen lassen. Nach dem Laden der gewünschten Datei wird das Bild angezeigt. Nun wird wiederum mit der Maus ein Ausschnitt des Bildes festgelegt oder, falls gewünscht, auch das ganze Bild übernommen. Nachdem diese Vorarbeit geleistet wurde, kann der Bildausschnitt zur weiteren Verwendung abgespeichert werden. Hierbei wird das Bild schon für die Aufnahme in einen bestehenden Text vorbereitet.

Bei unserem Test benutzten wir zur Bilderstellung DeluxePaint III. Die eben beschriebene Prozedur mit Auswählen, Speichern und Einsetzen des Bildes in den Text funktionierte dabei sehr gut. Nur auf eines muß man achten: die Bildformate sollten der Darstellung von UBM-Text entsprechen; den größten Erfolg hatten wir mit einer Zeichnung im 640 x 256 Pixelformat und vier Farben. Die auf der UBM-Diskette enthaltene Text-/Grafik-Datei entspricht ebenfalls diesen Anforderungen.

Ein mit 32 Farben digitalisiertes Bild wurde zwar ebenfalls problemlos eingelesen, da jedoch die Farbregister des Amiga anders belegt waren, wurde der Bildausschnitt farblich nicht so wiedergegeben, wie es hätte sein sollen.

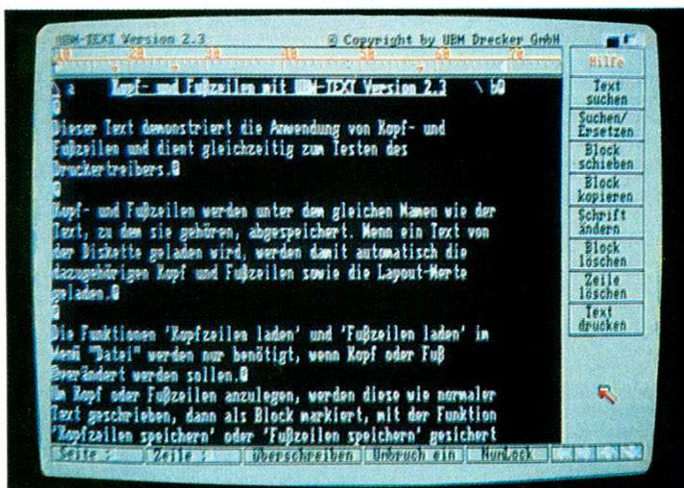


Bild 1: UBM-Text 2.3 bietet alles das, was man von einer guten Textverarbeitung erwarten kann. WYSIWYG und eine übersichtliche Benutzeroberfläche sind nur zwei von vielen positiven Punkten

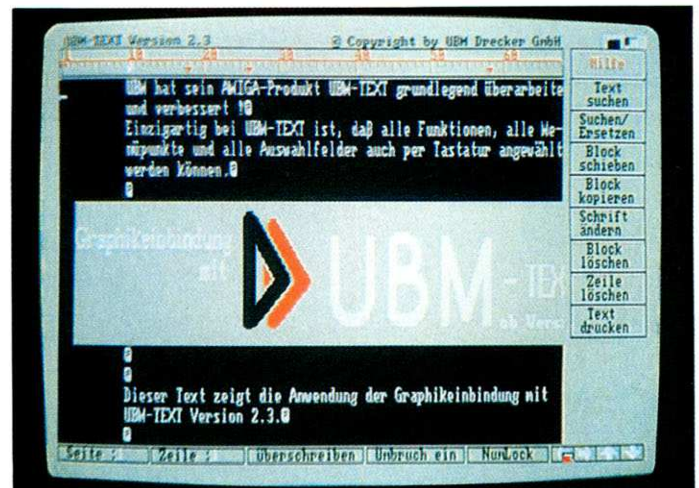


Bild 2: Die Einbindung von einfachen Grafiken in den Text geht bei UBM-Text ziemlich einfach vonstatten, wie man hier sehen kann. Ein Vorteil beim Sichten des Textes ist es, die Hilfsmenüs teilweise oder ganz ausblenden zu können



Bild 3: Zur Ausstattung des Programms gehört auch das 'UBM-Druck'. Hier besteht die Möglichkeit, den eigenen Drucker ohne größere Schwierigkeiten anzupassen. Tests können mit einem auf der UBM-Text-Diskette enthaltenen Probetext gemacht werden

Um UBM-Text gerecht zu werden, muß man klar und deutlich sagen, daß man es hier nicht mit einem Desktop-Publishing-Programm zu tun hat, sondern mit einer wirklich ausgereiften Textverarbeitung, die zudem auf die (eingeschränkte) Bildbearbeitung zurückgreifen kann. Daher muß man sich mit ein paar Einschränkungen zufriedengeben.

Zusammenfassung unseres Testes

UBM-Text ist eine Textverarbeitung, die gefällt. Nicht nur, daß die Texterstellung dem Benutzer einfach gemacht wird, auch die Möglichkeit, Bilddateien einzulesen und im Text unterzubringen, ist ein klarer Pluspunkt. Die beiden einzigen negativen Punkte, die wir finden konnten, sind:

a) die niedrige Geschwindigkeit und

b) die Schwierigkeit, UBM-Textdateien in andere Formate zu konvertieren.

Punkt a) wird dabei allenfalls Schnellschreibern auffallen, und mit Punkt b) läßt sich leben, wenn man nicht gerade Redakteur ist und Texte verschiedener Formate bearbeiten muß. Fazit: ein gut gelungenes Textprogramm.

(jb)

AMIGA DOS Blitzlicht

Name: UBM-Text 2.3

Hersteller: UBM-Drecker GmbH

Preis: 149,– DM

System: Alle Amiga mit mind. 512 kByte

Besonderheiten: Einbinden von Grafiken, WYSIWYG, eigene Druckertreiber, Ausschnitt-Programm für Grafik.

Positiv:
- übersichtliche Benutzerführung
- ausführliches Handbuch
- Darstellung dessen, was auf dem Drucker erscheint
- Funktionstasten mit Funktionen belegt (Schablone enthalten)
- Grafikeinbindung von IFF-Dateien

Negativ:
- Geschwindigkeit niedrig
- Textdateien im UBM-Format schlecht konvertierbar

Schneller geht's nicht...
Abbildung wurde digitalisiert
1/50 Sekunde
für ein professionelles Bild macht Schluß mit dem minutenlagen Warten.
● S/W-Videobilder in nur 1/50 s digitalisieren!
● Alle AMIGA-Grafikmodi werden unterstützt
● Auflösungen bis zu 704 x 552 Bildpunkten
● 2 bis 46 Grautöne oder 16 bis 4096 Farben
● Bis zu 7 Video-Eingänge, für alle Videoquellen
SNAPSHOT!® PRO (S/W-Echtzeitdigitizer) 895,-
SNAPSHOT!® RGB (Farb-Adapter für PRO) 395,-
SNAPSHOT!® Studio (Farb-Komplettgerät) 2375,-
Kostenlose Informationen anfordern!
... ein deutsches Spitzen-Produkt der VIDEOTECHNIK DIEZEMANN Dammstraße 42, 2300 Kiel 1, Telefon (0431) 9 44 24, Telefax (0431) 9 24 32

Einladung... ...ins Software-Paradies!
Riesenauswahl!
Wählen Sie aus über 5000 Programmen und über 2000 Computerbüchern Ihren Artikel.
In Selbstbedienung, oder mit fachkundiger Beratung, ganz wie Sie wünschen.
Software-Katalog!
Fordern Sie unseren kostenlosen Software-Katalog an!
● über 100 Seiten stark
● randvoll mit Programmen und Computerbüchern
Heute noch nebenstehenden Gutschein ausfüllen und einsenden.
SOFTSHOP
Ihr Computer träumt von uns!
4100 Duisburg 1, Sonnenwall 83
Tel. 0203/22409, Fax: 0203/29756
Gutschein für einen der umfangreichsten Software-Kataloge Deutschlands!
2/90 DMV
Kostenlos
Name:
Straße:
PLZ+Ort:
Telefon:
NEUE TOP-ANGEBOTE
☐ Leisure Suit Larry II 89,-
☐ Kaiser dt. 104,-
☐ Indiana Jones Adv. dt. 68,-
Datum Unterschrift
Sie möchten bestellen? Einfach ankreuzen, Absender eintragen, unterschreiben und an untenstehende Adresse schicken. Versand erfolgt per Nachnahme zuzüglich 6,- DM Versandkosten.
(Katalog alleine Versandkostenfrei!)
SOFTSHOP, Sonnenwall 83, 4100 Duisburg 1



Die erste Version von X-Copy war schon eines der fähigsten Kopierprogramme auf dem Amiga. Nun ist es auch möglich, Long-Tracks zu kopieren

X-Copy II

Sicherheits-Backups auch von kopiergeschützter Software

Viele Softwarehäuser haben, um den Raubkopierermarkt zu unterbinden, ihre Programme mit einem Kopierschutz versehen. Doch wenn man sich die verbreitete Virenplage auf dem Amiga ansieht, stellt sich unweigerlich die Frage, ob man wirklich seine Originaldiskette in das Laufwerk schieben soll. Um sich aber auch von kopiergeschützter Software Backups ziehen zu können, bedarf es spezieller Kopierprogramme.

Bodo Boxbeutel spielt gern Adventures. Abends, wenn er geschafft von seiner Arbeit kommt, macht er seinen Amiga an und schiebt die Diskette, auf der sein Lieblingsadventure ist, in den Floppyschacht, so auch heute. Doch plötzlich stutzt er: Ist sonst gewöhnlich schon nach wenigen Sekunden ein Titelbild erschienen, so meldet sich sein Rechner diesmal mit einem Requester, der besagt, die Diskette hätte einen Read/Write-Error. Der Grund dafür liegt darin, daß ein Virus zugeschlagen hat. Was tun, denn Wehklagen hat ja auch keinen Sinn.

Bodos Abenteuer mit kopiergeschützter Software

Bodo beschließt, am nächsten Tag zu seinem Fachhändler zu gehen, um das Programm umzutauschen. Gesagt, getan, doch der Händler weigert sich strikt, das Programm umzutauschen. Da das Programm kopiergeschützt war, hatte Bodo auch nicht die Möglichkeit, sich eine Sicherheitskopie anzufertigen. Bodo blieb nichts anderes übrig, als die Diskette zu formatieren und als Datendiskette zu benutzen – eine teure Leerdiskette.

Was hier wie ein Märchen klingt, ist in Wirklichkeit aber sehr oft der Fall. Viele Softwarehersteller versehen ihre Produkte mit immer besserem Kopierschutz, so daß es kaum möglich ist, von diesen Originalen Sicherheits-Backups zu ziehen. Alte Anwenderweisheit: Wenn es immer besseren Kopierschutz gibt, so wird es auch immer bessere Copy-Programme geben, um den Schutz zu umgehen.

X-Copy – nicht nur ein einfaches Kopierprogramm

An dieses Motto haben sich auch die Programmierer von X-Copy II angelehnt. Besonders viel Wert wurde dabei auf Benutzerfreundlichkeit gelegt. Voreinstellungen wie Start- und End-Track (bis Track 81), die Synchronisationslänge, das Quell- bzw. Ziellaufwerk (oder die Ziellaufwerke), die Diskettenseite sowie der Kopiermodus (dazu später mehr) und das Device sind durch einfaches Anklicken mit der Maus einstellbar. Die Synchronisationslänge stellt eine Markierung auf einem Track dar, an welcher das Leseprogramm erkennen kann, wo die zu lesenden Daten auf diesem Text beginnen. Will man sichergehen, daß der Kopiervorgang auch korrekt vonstatten geht, ist die Möglichkeit gegeben, eine Verifizierung vornehmen zu lassen. Dies ist jedoch nicht im NIBBLECOPY-Modus der Fall. Daß X-Copy nicht nur ein reines Kopierprogramm ist, beweist das Toolkit. In diesem Modus kann formatiert, der Name einer Diskette abgefragt, eine Disketten-Optimierung vorgenommen und eine Diskette nach Strukturfehlern untersucht werden.

Die einzelnen Tracks, die gerade "abgearbeitet" werden, werden in zwei Rasterfeldern angezeigt. Dabei werden "normale" Tracks mit einer grünen Null dargestellt. Sollte ein Track modifiziert sein (beispielsweise ein Kopierschutz), verändert sich die Anzeige des jeweiligen Sektors in eine rote bzw. blaue Zahl. Diese Zahl gibt Aufschluß über einen Diskettenfehler, der beim Lesen bzw. Schreiben aufgetreten ist.

Drei Kopierprogramme in einem

Ich hatte schon einmal erwähnt, daß es bei X-Copy II verschiedene Kopierarten gibt. Der erste Modus, DOSCOPY, ist für die "normalen" Amiga-DOS-Formate. Der zweite Modus, DOSCOPY+, ist schon etwas interessanter. Hier werden kleinere Diskettenfehler korrigiert. Der dritte Modus im Bunde namens NIBBLECOPY ist der interessanteste: Dabei wird auch ein eventueller Kopierschutz umgangen. Hierbei ist es dem Programm egal, wie die einzelnen Tracks beschaffen sind. Es wird versucht, ein möglichst genaues Duplikat der Originaldiskette zu schaffen. Leider scheitert NIBBLECOPY an Long-Tracks.

Dafür ist der Hardwarezusatz zuständig. Dieser wird zwischen den Diskettenport und das externe Laufwerk gesteckt. Das Programm, das diesen sogenannten Dongle unterstützt, ist jedoch nicht mit dem X-Copy II identisch, es befindet sich aber auch auf der Diskette unter dem Namen Cyclone II.

Ein Hardwarezusatz, der auch Long-Tracks kopiert – Kopierschutz ade?

War die erste Version von X-Copy schon ein leistungsfähiges Kopierprogramm, so erscheint die zweite Version noch fähiger und benutzerfreundlicher. Wer jedoch denkt, daß X-Copy II jede Software kopiert, der irrt. Beispielsweise Laserlöcher lassen sich auch mit diesem Programm nicht kopieren.

Es ist mit dem NIBBLECOPY-Modus auch möglich, Fremdformate (Atari ST, Acorn Archimedes und IBM) zu duplizieren.

Alles hat seine Schattenseiten, so auch X-Copy II: Besitzer einer 68020- bzw. 68030-Karte kommen nicht in den Genuß, mit X-Copy II arbeiten zu können. Hier spielt das Programm nicht mit.

(br)

AMIGA DOS Blitzlicht

Name: X-Copy II

Hersteller: Cachet

Preis:
ca. 50 DM (ohne Hardware)
ca. 70 DM (mit Hardware)

Positiv:
- übersichtliches Menü
- schnelle Kopier- bzw. Formatierroutinen
- Long-Tracks mit Hardwarezusatz kopierbar
- kopiert Fremdformate (PC, ST, Archimedes...)
- Tracks einzeln einstellbar (bis Track 81)

Negativ:
- läuft nicht mit 68020/68030-Karten

Professional Music Artist

Der Amiga ist berühmt für seine hervorragenden Soundmöglichkeiten. Um die Erstellung von Musikstücken so einfach wie möglich zu gestalten, gibt es sogenannte Sound- bzw. Musikeditoren, zumeist sind solche Eingabehilfen jedoch nicht gerade billig. Mit dem Professional Music Artist soll sich dies ändern.

Haben Sie nicht auch Lust, aus Ihrem Amiga so herrliche Musiken herauszuholen, wie man sie von vielen Spielen her kennt? Wenn Sie über genügend Programmierpraxis verfügen, sollte dies keine größeren Schwierigkeiten bereiten; Was aber macht derjenige, der die Programmierung nicht oder nur wenig beherrscht?

Nun, demjenigen sei auch geholfen, er benötigt lediglich einen Musik- bzw. Soundeditor. Wenn man sich jedoch die auf dem Markt befindlichen Produkte einmal ansieht, stellt man bedauerlicherweise fest, daß gute Software zumeist recht teuer ist. Mit dem Professional Music Artist (im folgenden kurz PMA genannt) von Michael Winterberg kann man sich nun einen Musikeditor zulegen, den jeder Geldbeutel verschmerzen kann. Nach einer kurzen Eingewöhnungsphase solllte es kaum Schwierigkeiten bereiten, eigene Musikstücke zu komponieren; man braucht noch nicht einmal Noten lesen zu können.

Musik en masse

Die Besonderheit am PMA liegt in erster Linie in der Abspieldauer der Musikstücke. Hier lassen sich Kompositionen bis zu einer Stunde Spieldauer erstellen, wobei bis zu 32 Instrumente ausgewählt und benutzt werden können. Es ist natürlich klar, daß die Spieldauer von der Abspielgeschwindigkeit und vom verbrauchten Speicherplatz abhängig ist. Der freie Speicher sowie die noch zur Verfügung stehenden Instrumentenplätze werden übrigens die ganze Zeit am Bildschirm angezeigt.

Von der Eingabe zur Musik

Wie übergibt der PMA die einzelnen Töne an den Rechner? Nun, jede einzelne Position, die für die Toneingabe relevant ist, beinhaltet vier sogenannte Patterns (für jede Stimme ein Pattern), wobei in jedem dieser Speicher bis zu 16 Noten (und deren Parameter) stehen können. Jede Position wird am Bildschirm angezeigt, in der die Eingabe als Hexadezimalwert getätigt werden kann. Der PMA unterscheidet bei der Eingabe zwischen zwei Editoren: zum einen den Pattern-Editor und zum anderen den Noten-Editor.

Im Pattern-Editor können zirka 6000 Patterns mit Sounddaten belegt werden. Diese Patterns können dann (nachdem man die Noten und deren Parameter in dem Noten-Editor gesetzt hat) beliebig in andere Patterns kopiert werden.

Im Noten-Editor werden, wie schon der Name sagt, die Noten editiert. Hier werden Einstellungen wie Oktave, Note, Attack, Decay und das Instrument für das jeweilige Pattern festgelegt. Der Attack-Wert steht dabei für das Anschwellen, der Decay-Wert für das Ausklingen des jeweileigen Tones, also ob der Ton schnell oder langsam an- bzw. abklingt.

Pulldown-Menüs für den schnellen Überblick

Haben Sie ein Pattern mit dem Noten-Editor erstellt, können Sie es beliebig oft kopieren (jedenfalls solange der Speicherplatz ausreicht). Gerade bei Baß- oder Schlagzeugthemen, die öfters wiederholt werden, findet diese Funktion optimalen Einsatz.

Der PMA kann sowohl über die Maus (mit Pulldown-Menüs) als auch per Tastatur bedient werden. Hier kann man unter anderem die einzelnen Patterns kopieren, Instrumente und Musikstücke einladen bzw. löschen etc. Ein besonderer Punkt liegt mit dem User-Menü vor. Mit der Funktion Optimize werden die im Speicher befindlichen Musikdaten auf die kürzeste Form komprimiert. Dann kann man sie (mit all ihren Instrumenten) auf Diskette abspeichern, um eigene Musikstücke in bestimmten Programmen zu verarbeiten.

Damit man beim Editieren einer Musik nicht zu lange mit dem Cursor herumfahren braucht, existiert eine Funktion, mit der man direkt zu einer Position springen kann. Weiterhin läßt sich in diesem Menü die Abspielgeschwindigkeit einstellen.

Ein weiterer Vorteil gegenüber einigen anderen Editoren liegt darin, daß man sich die Musikstücke während der Eingabe anhören kann. So kann man schnell und leicht Mißklänge feststellen und ausbessern. Zudem besteht die Möglichkeit, selbst erstellte Kompositionen in Demos einzubinden; eine auf der Diskette vorhandene Assembler-Abspielroutine macht's möglich.

Anhören während des Editierens möglich

Wenn Sie Instrumente von anderen Musikprogrammen (z.B. Ultimate Soundtracker) benutzen wollen, so müssen Sie lediglich darauf achten, daß sich diese Instrumente in einem Verzeichnis namens Instruments befinden müssen; Auch die Musikstücke müssen in einem Extra-Verzeichnis (Musik) stehen, damit sie vom PMA erkannt werden. Hier ist von Vorteil, daß der PMA auch mehrere Diskettenlaufwerke und Festplatten unterstützt.

Leider liegt bei dieser Version (1.0) noch keine Option zur Ausgabe auf einen Drucker vor, ist aber laut Hersteller in Planung. Eine weitere negative Eigenschaft liegt darin, daß bestimmte Fehlermeldungen nicht abgefragt werden und so in einem "Guru" enden.

(br)

Was auf den ersten Blick recht unübersichtlich aussieht, entpuppt sich nach relativ kurzer Einarbeitungszeit als einfach zu handhabendes Werkzeug zur Musikeingabe

AMIGA DOS Blitzlicht

Name: Professional Music Artist
Vertrieb: Digital Marketing
Preis: 34,95 DM

Positiv:
- niedriger Preis
- max. Abspieldauer bis zu einer Stunde
- bis zu 32 Instrumente einsetzbar
- Musiken in eigene Demos implementierbar
- Anhören der Sounds während des Editierens

Negativ:
- gewöhnungsbedürftige Eingabe
- manche Fehlermeldungen nicht abgefragt

MUSIK IM BYTE

SIDMON – Supersounds im Eigenbau

Wer Amiga sagt, muß auch Musik sagen. Gerade weil der Amiga über einen der leistungsfähigsten Soundchips verfügt, die jemals in Computer eingebaut wurden, erwartet man mit Spannung Neuigkeiten in dieser Beziehung. Die neue Software-Firma 'Turtle Byte' will mit ihrem Erstlingswerk in diesem Genre 'SIDMON'-Maßstäbe in der Sound-Erstellung setzen. Wir haben uns für Sie einmal näher mit dem Programm beschäftigt.



Wenn man sich im Augenblick unter den Sound-Programmen auf dem Amiga umschaut, kristallisiert sich am Schluß ein Name heraus, der wohl ein Maßstab für alle weiteren Programme ist: Der 'Soundtracker' von EAS Software. Dieses Sound-Editierungs-Programm gab ein positives Beispiel für das Weglassen einer Noteneingabe und die dafür vorkommende Pattern-Nutzung. 'SIDMON' will nun Platz Eins erreichen. Ob es ihm gelingt? Wir wollten es wissen.

Ich 'sample', du 'samplest – SIDMON und die digitalisierten Töne

Der Soundeditor SIDMON macht sich eine der besten Eigenschaften von 'Paula' – dem Soundchip des Amiga – zu Nutze, indem er auf digitalisierte Instrumente zurückgreifen kann. Auf der Sample-Diskette von beiden 'SIDMON'-Datenträgern befinden sich daher eine ganze Menge Instrumente, die man sich nach Belieben zusammensuchen kann. Wem das nicht genügt, dem bietet SIDMON eine Besonderheit: Frei einstellbare Wellenformen, die der Erzeugung eigener Instrumente dienen. Damit ist einer der Pluspunkte von SIDMON schon genannt: Das Programm simuliert einen Synthesizer – eine Eigenschaft, die 'Paula' entgegen anderslautender Meinungen so nicht beherrscht.
Wie aber nutzt man SIDMON nun richtig? Um zu selbsterstellten Sounds zu gelangen, gehört eine gewisse Ordnung in die Arbeit mit diesem Sound-Programm. Angefangen wird daher mit dem

Instrument-Editor – Erste Zwischenetappe für kleine Mozarts

Der Instrumenten-Editor ist sozusagen die erste Anlaufstelle auf dem Weg zum Hit. Hier gibt es zwei Möglichkeiten, um die Lautsprecher des Monitors, oder noch besser der heimischen Stereo-Anlage, zum Erklingen zu bringen:
Man begibt sich direkt zum Sample-Menü, um die vorge-

fertigten, digitalisierten Sounds (wir werden sie im weiteren ebenfalls Samples nennen) zu laden und anzuhören. Die Sounds, die man benötigt, müssen nun im Instrumenten-Editor einer Instrumenten-Nummer zugeordnet werden. Und hier wird es ein bißchen kompliziert. Wie schon gesagt, stellt SIDMON dem Anwender einen Synthesizer zur Verfügung, wo mit vorgefertigten oder selbsterstellten Wellenformen gearbeitet werden kann. Diese Wellenformen werden ebenfalls im Instrumenten-Editor verwaltet, sie belegen die 'Waveform'-Nummern 1 bis 15. Die Instrumente können nach Anwahl der Instrument-Nummer mit der jeweiligen Wellenform oder einem Sample belegt werden. Ein kurzes Beispiel:

Instrument = 1:Waveform No. = 1 –

Das Instrument 1 wird mit einem selbsterstellten Synthesizer-Ton belegt;

Instrument = 1:Waveform No. = 16 –

Jetzt wird Instrument 1 mit der ersten Sample-Nummer belegt, also dem Sample, der unter der Nummer 16 geladen wurde.

Am Anfang mag einem diese Nummernvielfalt etwas verwirrend vorkommen, wenn man allerdings SIDMON zwei-, dreimal geladen hat, ist die Gewöhnungszeit vorbei, und man lädt Samples oder editiert Wellenformen ohne Probleme. Etwas mehr Schwierigkeiten macht im Sample-Menü die Eingabe des Sample-Starts, des Loop-Starts und -Endes. Hier muß man den geladenen Sample oder den erstellten Platz im Speicher reservieren. Mit 'LOOP-START' und 'LOOP END' kann außerdem ein Teil des Samples bestimmt werden, der sich nach einmaligem Abspiel immer wiederholt. Hier bietet die Funktion 'SEARCH LOOPEND' eine wertvolle Hilfe. Innerhalb des Samples wird der nächste 'Hochpunkt' der Wellenform gesucht und angezeigt, danach braucht man im Instrumenten-Editor nur noch wenige Korrekturen im 'FINE TUNING' zu machen, und schon hat man den perfekten Sample-Loop im Amiga.

Insgesamt stellt SIDMON 200 kByte für Instrumente zur Verfügung. Da aber jeder Sample nicht gerade platzsparend ist, muß man hier darauf achten, daß der Platz 'gerecht' aufgeteilt wird. Deshalb sollte man sich zuerst mit den Funktionen der Sample-Menüs vertraut machen.

Hat man alle Samples, die man benötigt, zusammengestellt und eine Reihe von Instrumenten belegt, ist es sinnvoll, die weiteren Funktionen des Instrumenten-Editors zu probieren. Hier findet man außer verschiedenen Soundfunktionen wie Einstellungen der ADSR-Kurve (Attack, Decay, Sustain und Release, sehen Sie dazu auch unseren Soundkurs), die Möglichkeit, den Klang durch die Arpeggio-Funktion zu verändern. Ein Arpeggio setzt sich aus 16 sehr schnellen Veränderung der Tonhöhe zusammen. Um sich dies vorzustellen, ist es eigentlich notwendig, Ergebnisse dieser Funktion zu hören, die sich aus den beiden Unterfunktionen 'ARPEGGIO BYTE COUNT' und 'ARPEGGIO BYTE' zusammensetzt. Dabei gibt 'ARPEGGIO BYTE COUNT' die Tonhöhenveränderung an, die mit 'ARPEGGIO BYTE' zugewiesen wird. 'PHASE-SHIFT' und 'PHASE-SPEED' haben es ebenfalls in sich. Mit 'PHASE-SHIFT' wird eine WAVEFORM-Nummer ausgewählt, mit 'PHASE SPEED' wird in 1/50 Sekunden der Einsatz des Phase-Effekts nach erklingen des Tones eingestellt. Was aber ist der Phase-Effekt? Er läßt einen Ton ständig um eine Note schwingen. Besonders gut läßt sich dieser Effekt bei Sinus-Kurven hören, da hier der Schwingungston ebenfalls als Sinus den eigentlichen Ton auf- und abschwellen läßt.

Der Pattern-Editor – Stück für Stück zum Computerhit

Mit der 'PITCH FALL'-Funktion läßt sich ein Ton ständig in der Tonhöhe verändern, ebenfalls eine gut nutzbare Möglichkeit, wenn man mit eigenen Wellenformen arbeitet.

Weiterhin bieten sich die Mixfunktionen an, sie erlauben das Mischen von Wellenformen in vielfältiger Art. Mit 'WAVE-PRESET No.' kann eine der vorgefertigten Wellenformen für ein Instrument übernommen werden. Damit sind wir bei der eigentlichen Überraschung des Instrumenten-Editors: dem Sample-Anzeigefeld. Es sieht aus wie ein integriertes Oszilloskop, und so ist auch seine Funktion: Die aktuelle Sample-Waveform wird hier angezeigt. Um dem Ganzen die Krone aufzusetzen, bietet sich hier das Erstellen einer eigenen Wellenform mit der Maus an. Einfach ins Editierfeld gehen und die Wellenform zeichnen, mehr ist nicht nötig. Um ehrlich zu sein, hier sollte man ein bißchen Verständnis für Wellenformen und die durch sie erzeugten Töne haben, da es sonst zu recht seltsamen Geräuschen kommen kann. Aber auch hier gilt: Übung macht den Meister. Und eine ruhige Hand ist dabei auch nicht zu verachten.

Instrumente sind ja gut und schön, aber ohne Komposition ist das schönste Instrument uninteressant. Ob man nun eigene Songs erstellen möchte oder bekannte Hits aus den aktuellen Top Ten (nicht öffentlich vorspielen, gelle? Schönen Gruß an die GEMA), beim Pattern-Editor trennt sich die Spreu vom Weizen. Im Pattern-Editor kann man mit der Pattern-Nummer das zu bearbeitende Pattern auswählen. Eine 'PLAY'-Funktion ermöglicht das gleichzeitige Mithören der Töne, sie kann vor dem Editieren angeklickt werden. Im Pattern muß natürlich das benutzte Instrument angegeben werden, sonst hört man nichts. Der Pattern-Editor verfügt allerdings auch über besondere Editiermöglichkeiten, so läßt sich ein fließender Übergang von der einen in die andere Note herstellen, und Geschwindigkeit sowie Lautstärke jeder einzelnen Note lassen sich beeinflussen.

Das Pattern verfügt über insgesamt 64 Einzelnoten, hat also eine Unterteilung in 64 Abschnitte. Diese Länge läßt sich ebenfalls variieren, so daß nicht die volle Patternlänge gespielt werden muß. Von Vorteil kann dieses vor allem dann sein, wenn man nicht auf einen in 64 Einzelschritte aufteilbaren Takt zurückgreifen will.

Bei Pattern-Editierung muß man eigentlich ein gutes Gefühl für Rhythmus besitzen, sonst kann es schnell passieren, daß man total aus dem Takt kommt, sowohl mit der Musik als auch mit der Stimmung. Dies gilt nicht nur für SIDMON, sondern auch für andere Musikprogramme, die auf Pattern-Erstellung zurückgreifen.

Der Song-Editor – Die Stunde der Wahrheit: Hit oder Flop?

Im Song-Editor werden die Pattern schließlich zu einem ganzen Musikstück zusammengefaßt. Hier findet man ebenfalls elementare Unterschiede zum bekannten Soundtracker: Songs bestehen bei SIDMON nicht aus Patterns, sondern aus Sequenzen, die wiederum aus Patterns bestehen. Der Sinn ist ziemlich schnell klar: Wenn ein Pattern mehrmals vorkommen soll, ist es unsinnig, dieses auch mehrmals zu editieren. Statt dessen kann man in den Sequenznummern das jeweilige Pattern auf einen der vier Kanäle legen, in der nächsten Sequenz dieses beibehalten, aber auf ei-

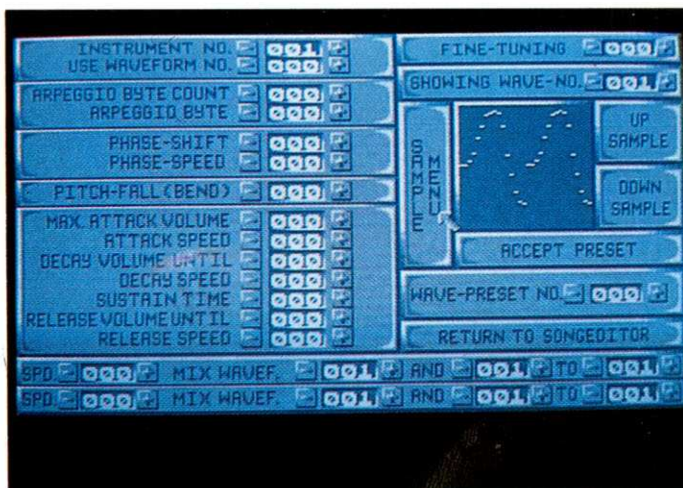


Bild 1: Eine der ungewöhnlichen, aber sehr nützlichen Funktionen ist das Erzeugen eigener Wellenformen, die die Ausgangsbasic für Töne bilden

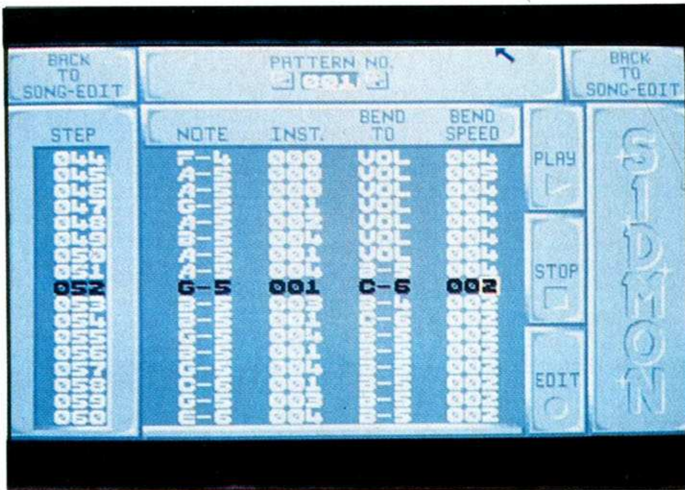


Bild 2: Das Sample-Menü. Hier können fertige Samples geladen werden. Etwas ungewöhnlich ist die geforderte Eingabe der Start-und Loop-Werte, jedoch wird man dies schnell schätzen lernen.

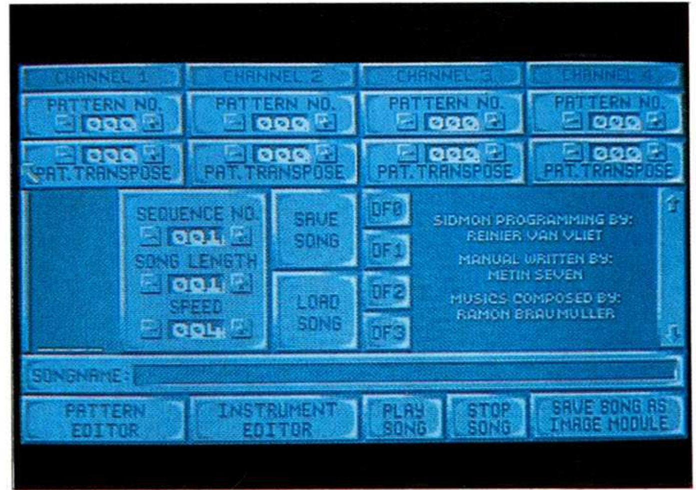


Bild 4: ...die schließlich im Song-Editor zu einem kompletten Stück zusammengesetzt werden. Die Pattern werden allerdings erst zu Sequenzen zusammengefaßt, wobei jede Sequenz alle vier Soundkanäle beeinflussen kann

nem anderen Kanal ein zweites Pattern abspielen.

Die Summe der Sequenzen ergeben dann das gesamte Musikstück. Im Song-Editor ist es dann auch möglich, die erstellten Songs in zwei Variationen abzuspeichern:

○ Als IMAGE MODUL speichert man dann ab, wenn man das Musikstück in eigene Programme einbinden möchte, denn SIDMON speichert das File als Maschinencode ab.

○ Die zweite Möglichkeit ist wohl die wichtigste, denn 'SAVE SONG' speichert den Song als SIDMON-File ab, und nur als dieses ist es wieder ladbar.

Einbinden lassen sich die IMAGE-Files unter allen bekannten Assemblern, Assembler-Kenner sind hierbei eindeutig wieder im Vorteil, aber auch nur, was die Nutzung von SIDMON-Musikstücken unter Fremdprogrammen angeht.

Um es kurz zu machen, SIDMON ist auf dem besten Wege, sich in der Redaktion der AMIGA DOS beliebt zu machen. Wenn man sich erst ein bißchen in das Programm eingearbeitet hat, kommt man so bald nicht wieder davon los.

Kann SIDMON halten, was es verspricht? Wir meinen: aber sicher!

Es ist wirklich verwunderlich, was man dem Amiga alles an Tönen entlocken kann, und das sogar mit dem Hintergrundwissen, daß dieser Computer an sich schon über einen Super-Soundchip verfügt. Darüber hinaus kann SIDMON auch Fremd-Samples benutzen, wenn diese im IFF-Format vorliegen, ein weiterer Grund für Zufriedenheit. Turtle Byte kündigte außerdem an, daß es eine erweiterte Version geben wird, die über eine MIDI-Funktion verfügen wird, man darf also weiterhin mit SIDMON rechnen. Benutzer der jetzt vorliegenden Version werden, wenn sie ihre Registrierungskarte ausgefüllt und an Turtle Byte geschickt haben, von Turtle Byte benachrichtigt und können die ältere Version gegen die neue und einen geringen Aufpreis eintauschen. Ein Service, der leider nicht immer selbstverständlich ist. Alles in allem kann man SIDMON also sehr empfehlen.

An wen richtet sich SIDMON nun? Eigentlich an jeden Amiga-Benutzer, denn auch wenn man keinerlei Kenntnisse in Assembler hat, lassen sich mit SIDMON tolle Sounds erstellen und abspielen, vorausgesetzt, die Musikalität endet nicht nach der zweiten Strophe von 'Hänschen klein'. Und die Profis (sowohl Musik- als auch Programmier-Profis) werden wohl am meisten ihre helle Freude am SIDMON haben, braucht man sich hiermit doch nicht mehr die Gehirnwindungen im Programmieren von Soundfunktionen zu verrenken. Fragt sich also zuletzt, was man negativ bewerten kann? Wir meinen, eigentlich nichts. Ein paar Schwachpunkte des Handbuches, die vor allem Neueinsteigern beim Erlernen etwas Schwierigkeiten machen können, sind nicht als negativ, höchstens als verbesserungswürdig einzustufen. Und deshalb unser Gesamturteil: sehr gut.

(jb)

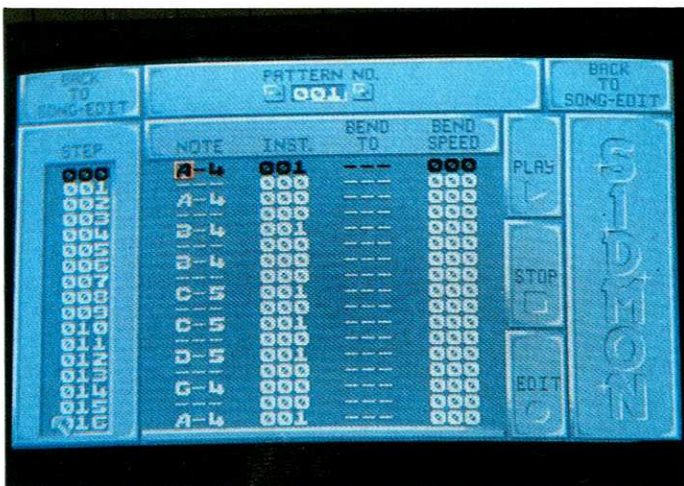


Bild 3: Hat man sich für einige Sample-Sounds entschieden, geht es ans Komponieren. Der Pattern-Editor dient zum Erstellen von Teilstücken,...



AMIGA DOS
Blitzlicht

Name: SIDMON
Hersteller: Turtle Byte
Vertrieb: Fachhandel
Preis: 89,- DM

Positiv:
- Erstellen eigener Wellenformen möglich (bis zu 15 Stück)
- Synthesizer-Simulation
- übersichtliche Editoren
- arbeitet auch mit IFF-Samples
- Speicherung der Songs als Maschinencode integriert
- ADSR-Kurven jedes Instrumentes beeinflußbar
- Mischen von Wellenformen
- weitere Sampledisketten erhältlich
- Update Services
- sehr gutes Preis-/Leistungs-Verhältnis

Negativ:
Keine Negativpunkte erkennbar, dafür einen Schwachpunkt:
- Anleitung an manchen Stellen nicht ausführlich genug

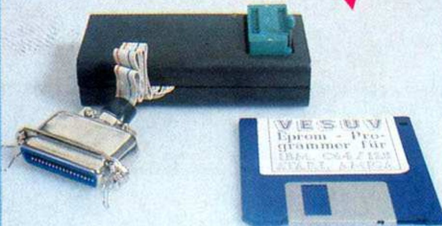


VIRUS-FALLE **29,95 DM**
verhindert das Ausbreiten von Boot-Viren.

A 512 **179 DM**
512 K Speichererweit. ⊕ abschaltbar ⊕ Uhr

LIGHTPEN-Mouse **99 DM**
Mauskompatibel: Sie können **direkt** auf dem Monitor zeichnen

LIGHTPEN ohne Maustasten **79 DM**

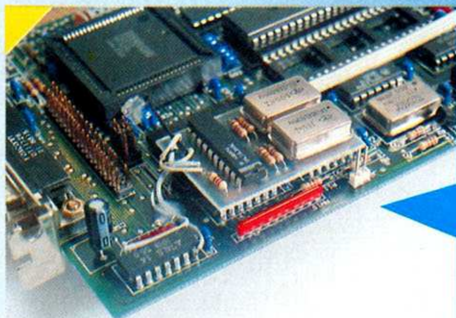


VESUV-AMIGA-Eprommer 199 DM
läuft auf A500, 1000 und 2000 ⊕ brennt auch 1 MBit-Eproms ⊕ „HAPPY" 3/89 Test-Gesamturteil „SEHR GUT" ⊕ programmiert die Eproms 2716-27512, 27513 und 27011

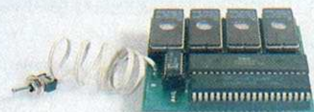


MEGA-DRIVE **299 DM**
2MByte Diskettenlaufwerk ⊕ 1,4 MByte formatiert ⊕ arbeitet auch mit Ihren alten 720-K Disketten ⊕ abschaltbar ⊕ durchgeschleifter Bus

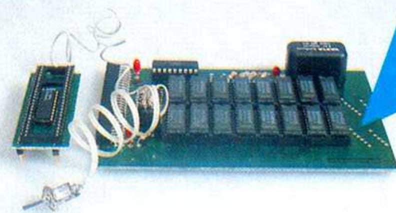
10 HD-Disketten (1,4 MB) **29,95 DM**



TURBO-AT voraussichtl. **398 DM**
TURBO-XT **199 DM**
⊕ Formel 1 für Ihre AT/XT-Karte
⊕ AT/XT-Karte nicht im Lieferumfang

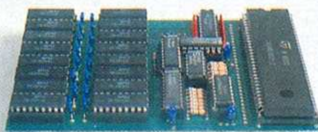


KICKSTART 3 **59,95 DM**
Umschaltplatine für 3 Kickstart
⊕ 2x original Kickstart-Roms und 1x Kickstart und 1x in Eproms (4x 27512)

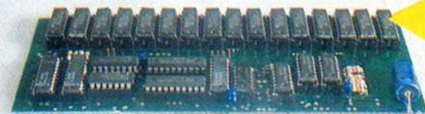


A2MB **598 DM**
2 MByte Ramkarte ⊕ mit FAT-AGNUS 1,8 MByte, mit BIG-AGNUS volle 2 MByte (Chipram/Fastram)





A8MB/1000 **798 DM**
8MByte Ramkarte; mit 2MByte bestückt ⊕ einfachster Einbau ⊕ kein Löten - nur einstecken ⊕ abschaltbar ⊕ ohne Waitstates



A8MB/2000 **698 DM**
8MByte Ramkarte mit 2 MByte bestückt ⊕ zukunftssicher durch 4-MBit-Technologie ⊕ autokonfigurierend ⊕ 0-Waitstates ⊕ abschaltbar ⊕ Anschluß für Reset-Taster

Ramkarte o. Abb.
mit 8 MByte bestückt **1998 DM**

Für 5 DM in Briefmarken erhalten Sie für unsere Amiga-Produkte eine Bedienungsanleitung und einen Wertgutschein (5 DM werden beim Kauf angerechnet).

...bei uns nutzen Sie heute Technologie von morgen

FOTAMIGA-FOTOLABOR
Ein Roboter entwickelt automatisch Ihre Color-negativ-Filme im C41-Prozeß ⊕ S/W-Film-Entwicklung ⊕ Color-Dia im E6-Prozeß auf Anfrage ⊕ elektronische Temperaturregelung ⊕ mit Programmdiskette

Fertige Elektronik mit Mechanik-Bausatz **698 DM**
Komplettes Fertiggerät **898 DM**

WELTNEUHEIT

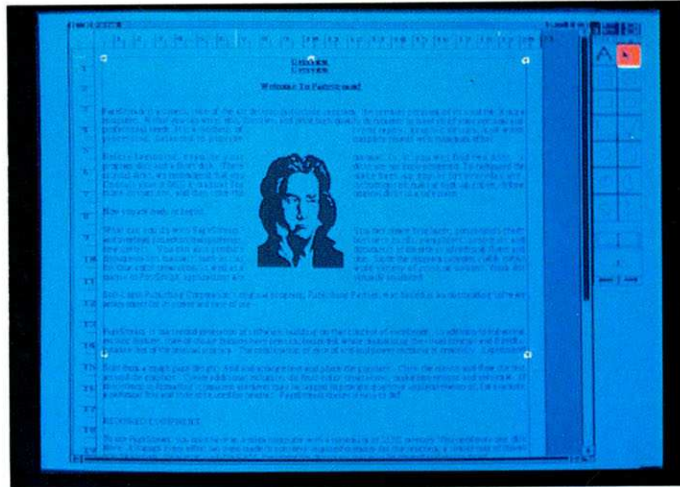


HOTLINE

Technische Fragestunde:
Mo.-Fr. von 16-17 Uhr. Hier können Sie die Entwickler unserer Amiga-Produkte sprechen.

0 22 25/20 61-20 62-20 63

Neuer Markt 21 5309 Meckenheim Telefon 0 22 25/20 61-62-63



Mit Pagestream steht dem Amiga-Besitzer ein DTP-Programm zur Verfügung, das durch seinen geringen Preis auch im Heimbereich Anwendung finden soll

Klaus Rupp

DTP auch für zu Hause

Ein Stern unter DTP-Programmen?

Programme für Desktop Publishing konnte man bisher auf dem Amiga in die Kategorien "gut & teuer" bzw. "billig & kaumzugebrauchen" unterteilen. Mit Pagestream schickt sich nun ein Programm an, den Konkurrenten das Fürchten zu lehren. So besitzt das Programm Pagestream-Funktionen, die man anderswo vergeblich suchen kann.

Zum Lieferumfang von Pagestream gehören ein umfangreiches Handbuch (zur Zeit leider nur in Englisch), die Programmdiskette und die Fonts auf einer Extradiskette. Sehr lobenswert ist, daß die Herstellerfirma Soft-Logik auf einen Kopierschutz verzichtet hat und sich das Programm problemlos auf einer Harddisk installieren läßt.

Die vom Hersteller angegebene Konfiguration von 512 kByte und einem Laufwerk sollte man als absolutes Minimum ansehen, besser und auch häufig vonnöten sind mindestens 1 MByte, wenn man mit dem Programm vernünftig arbeiten will.

Nach dem Starten des Programms beginnt man am besten mit der Festlegung der Voreinstellungen. Sehr wichtig ist hier die richtige Einstellung des Druckers, soll die volle Leistungsfähigkeit des Programms ausgenutzt werden. Weiterhin lassen sich (insofern Sie Pagestream auf einer Harddisk installiert haben) die Suchpfade in einem Extra-Requester ganz einfach und bequem vom Programm aus einstellen. Sind die Voreinstellungen beendet, und wird ein neues Dokument angelegt, werden vom Programm die Maße des Bearbeitungsfeldes abgefragt. Hier kann man die Dokumentgröße (Format A4, A5 ...) festlegen.

Der Menüführung von Pagestream muß ein Lob ausgesprochen werden, denn die einzelnen Menüpunkte sind nicht nur mit der Maus, sondern auch mit Hilfe der ESCAPE-Taste anwählbar.

Die Geschwindigkeit, mit der man den Mauszeiger durch die Menüs rollen kann, ist beachtlich. Der Konkurrent Professional Page schläft hierbei fast ein.

Einbindung von Text und Grafik: Wie sieht's aus?

Daß man bei einem DTP-Programm die Einbindung von Text und Grafik voraussetzen darf, versteht sich natürlich von selbst. Die Grafikimplementierung gestaltet sich bei unserem Testprodukt völlig problemlos. Bilder können auch im nachhinein beliebig verschoben, verkleinert oder vergrößert werden.

Es fällt auf, daß das Programm die Darstellung auf dem Bildschirm recht rustikal vornimmt. Das Ergebnis (und somit auch das endgültige Aussehen des Dokumentes) wird erst beim Ausdrucken berechnet. Dies ist jedoch nicht unbedingt als Nachteil anzusehen, weil durch diese Art der Darstellung die Arbeitsgeschwindigkeit um einiges erhöht wird. Um die Geschwindigkeit gerade beim Neuaufbau einer Seite noch zu steigern, kann man zusätzlich verhindern, daß das Programm Grafiken anzeigt. Anstelle der Grafiken erscheint dann nur ein Rahmen.

Als Besonderheit, die bisher nur bei Pagestream zu finden ist, kann man die Funktion ansehen, mit der es möglich ist, Texte in Boxen um ihre Achsen (X-,Y- und Z-Achse) zu drehen und zu spiegeln. Den Textfluß (um Grafiken herum) beherrscht Pagestream ebenfalls, wobei der Abstand zwischen Grafik und Text frei definiert werden kann.

Als weitere Besonderheit bei Pagestream ist zu erwähnen, daß gerade der Textfluß auch bei runden oder unregelmäßigen Objekten sich immer im richtigen Abstand anpaßt.

Eine unangenehme Begleiterscheinung trat jedoch bei der Implementation von Texten mit Umlauten auf. Diese werden entweder einfach verschluckt oder durch irgendwelche Zeichen ersetzt. Wobei wir auch schon beim Lexikon wären: Es ist bisher leider nur in Englisch verfügbar, was natürlich auch Auswirkungen auf die Trennungsautomatik hat. Schlicht gesagt, diese sollte man im Augenblick vergessen, aus dem einfachen Grunde, da sie nicht funktioniert.

Was hat der Font des Vehikels mit Buchstaben zu tun?

Die Zeichensätze des Programms arbeiten nach dem Prinzip der Vektorfonts, das bedeutet, daß nur bestimmte Punkte festgelegt werden. Die dazwischenliegenden Punkte werden dann vom Programm berechnet. Dies hat gegenüber "normalen" Fonts zwei entscheidende Vorteile:

Erstens ist man nicht auf eine bestimmte Größe eines Zeichensatzes festgelegt, sondern kann den Font auf verschiedene Größen bringen, die vom Programm her gestattet sind. Zweitens (und das ist für alle Besitzer von Nadeldruckern die Nachricht!) kennt das Programm durch die Berechnung der Fonts beim Ausdrucken nicht mehr das berüchtigte "Treppensyndrom". Alle Buchstaben und Sonderzeichen, in welcher Form und Größe auch immer, werden in einer Qualität auf Nadeldruckern wiedergegeben, wie sie bisher bei DTP-Programmen nicht möglich waren.

Einziger, aber verschmerzlicher Nachteil ist, daß Pagestream zur Berechnung und zum Ausdruck doch einiges an Zeit braucht.

Der private Zeitungsmacher

Jeder DTP-Freak stellt sehr bald fest, daß das Programm trotz des fehlenden deutschen Lexikons, der langen Ladezeit und des hohen Speicherplatzbedarfs zur Creme der DTP-Programme zu rechnen ist. Insbesondere das Ergebnis, also das fertige Dokument, rechtfertigt diese Einstufung des Programms, besonders wenn man nur über einen Nadeldrucker, gleich ob 9- oder 24-Nadler, verfügt. Setzt man dann noch den günstigen Preis von 398,– DM zugrunde, ist das Programm im Augenblick als konkurrenzlos anzusehen und schlägt den direkten Konkurrenten Professional Page nicht nur im Preis um Längen, sondern auch in der Leistung.

(br)

AMIGA DOS Blitzlicht

Name: Pagestream
Hersteller: Soft-Logik
Quelle: Fachhandel
Preis: 398,- DM

Positiv:
- geringer Preis
- Unterstützung von 9-Nadel-Druckern
- kein Kopierschutz
- einfache Menüführung

Negativ:
- relativ langsam
- Handbuch nur in Englisch

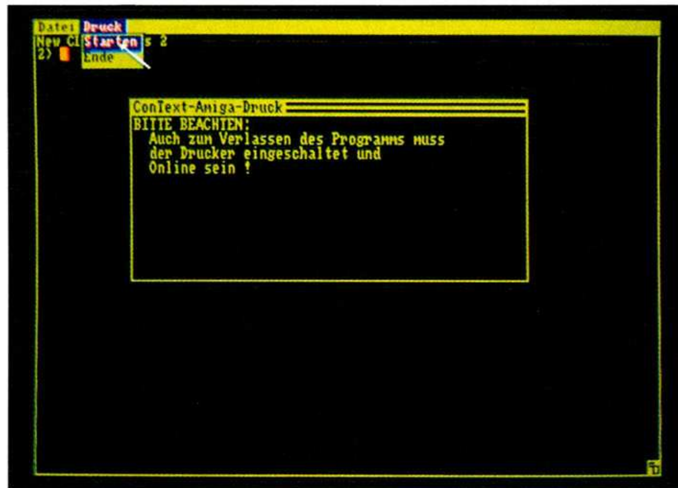


Bild 1: Bei ConText-Amiga-Druck ist es wichtig, daß der Drucker auch bei Beenden des Programms 'On-Line' geschaltet ist

Martin Schlöter

ConText lernt drucken!

Das Druckprogramm für ConText-Amiga

Was nützt einem die schönste Textverarbeitung, wenn man seine Texte hinterher nicht schwarz auf weiß wiederfindet? Also auf und in die Hände gespuckt, wir steigern den Wert von ConText-Amiga und verpassen ihm einen Ausgang Richtung PRT:.

Beim Amiga hat das Wort 'drucken' eine ziemlich schwierige Stellung. Dank des aufwendig programmierten Betriebssystems bieten sich dem Amiga-Besitzer hier immer wieder Möglichkeiten, einen Tag, der gut angefangen hatte, mit mieser Laune zu beenden. Irgendwie scheint es unser Wundercomputer mit dem Druckeranschluß zu haben, denn stets tauchen irgendwelche Probleme mit dem Druckertreiber, dem jeweiligen Programm oder überhaupt auf. Also heißt die Devise: Selbst programmiert der Mann (natürlich auch die Frau), oder man läßt programmieren. Sie haben Glück, wir haben für Sie die zweite Möglichkeit gewählt.

Der K(r)ampf mit dem Drucker oder 'ConText-Druck' kommt langsam, aber gewaltig!

'ConText-Druck' ist ein eigenständiges Programm und braucht demzufolge nicht unbedingt unter ConText- Amiga zu laufen. Andererseits ist es aber auch möglich, ConText-Druck als zweites Programm laufen zu lassen – schließlich sollte man die Fähigkeit des Amiga, im Multitasking-Betrieb arbeiten zu können, auch mal ausnutzen, nicht wahr?
Nach dem Start von ConText-Druck meldet sich dieses mit einem Text-Window, in dem ein wichtiger Hinweis steht:

Der Drucker muß während des Programmlaufs und beim Verlassen des Programmes auf 'On-Line' geschaltet sein.

Wenn Sie dieses beherzigen, besteht jetzt die Möglichkeit, mit der rechten Maustaste ein Menü aufzurufen. Eigentlich ist es nur ein kleines Menü. Es besteht aus den Wahlfunktionen LADEN (der Text, den Sie ausdrucken wollen. Er kann als ConText-, Layout- oder ASCII-Text vorliegen), QUIT (benötigt wohl keiner näheren Erklärung), STARTEN (gilt für den Druckbetrieb) und BEENDEN (ebenfalls eine Anweisung an den Drucker).
Nach dem Laden eines Textes, was mittels eines Directory-Windows geschieht, wird der Druckvorgang mit STARTEN (na, was wohl?) gestartet. Dabei wird die an den Drucker geschickte Zeile angezeigt. Nun kann es sein, daß Sie zu den Glücklichen gehören, deren Drucker über eine geraume Menge an Puffer-RAM verfügt. Dann dauert es ein bißchen, bis die erste Zeile schwarz auf weiß zu sehen ist. Sollte der Text auf einer Seite Platz haben, so meldet Ihnen das Programm dies mit der Bitte, doch endlich eine Taste zu drücken, um mit dem Drucken anfangen zu können. Falls Ihnen der Ausdruck nicht gefällt, gibt es immer noch die Möglichkeit, mit der Maus die Funktion BEENDEN anzuwählen. Dann stoppt das Programm, während Sie den Drucker stoppen müssen.
Damit wäre im großen und ganzen alles gesagt, nur noch eines: Das Programm ist in der vorliegenden Form für 24-Nadel-Drucker gedacht. In Abbildung 1 sind jedoch Zeilen des Programmes für 9-Nadel-Drucker umgeschrieben worden. Sie brauchen nur anstelle der im Programm enthaltenen eingesetzt zu werden. Anhand der Zeilen sollte es jedem möglich sein, für seinen Drucker im Handbuch die richtigen Steuercodes zu finden. Sollten Sie noch Anregungen oder Verbesserungsvorschläge haben, schreiben Sie uns doch mal. Wir werden Ihre Vorschläge gerne aufgreifen.

(jb)

An dieser Stelle, liebe Leser, werden wir in unregelmäßigen Abständen Tips und Tricks zum ConText-Amiga-Programm aus Heft 1'90 bringen. Die ConText-Ecke soll Utilities und Hilfestellungen zum Textprogramm beinhalten. Wenn Sie zum Gesamtprogramm ebenfalls Verbesserungsvorschläge oder Zusatzprogramme haben, melden Sie sich unter dem Stichwort "ConText-Amiga"!

```
100 ' ----- DRUCKERTABELLE -----
101 esc$=CHR$(27)
102 init$=esc$+"x"+CHR$(1) ! NLQ-Schrift
103 reset$=esc$+CHR$(64)
104 fettan$=esc$+"E"
105 fettaus$=esc$+"F"
106 unterstrichenan$=esc$+"-"+CHR$(1)
107 unterstrichenaus$=esc$+"-"+CHR$(0)
108 subscriptan$=esc$+"S"+CHR$(1)
109 subscriptaus$=esc$+"T"
110 superscriptan$=esc$+"S"+CHR$(0)
111 superscriptaus$=esc$+"T"
112 kursivan$=esc$+"4"
113 kursivaus$=esc$+"5"
114 cr$=CHR$(13)  ! Wagenruecklauf
115 lf$=CHR$(10)  ! Linefeed
116 ff$=CHR$(12)  ! Seitenvorschub
117 linkerrand$=esc$+"l"  ! Seitenrandeinstellung
118 rechterrand$=esc$+"Q"+CHR$(80) ! Rechter Rand 80 vom Linke
    n
119 zeilenproseite$=esc$+"C"
```

Abb.1: Für Besitzer von 9-Nadel-Druckern haben wir hier die Zeilen mit den entsprechenden Zeilencodes stehen. Diese Zeilen müssen im Programm ausgetauscht werden

```
' ***********************************************************
' *             ConText-Amiga-Druck  Version 1.0            *
' *     Druckprogramm zur WYSIWYG-Textverarbeitung          *
' *                     ConText-Amiga                       *
' *                                                         *
' *        (C) 1989,1990 Amiga-DOS & M.Schloeter            *
' ***********************************************************
'
' ***** Voreinstellungen *****
$&3
$*&
~SetTaskPri(FindTask(0),4)
CLEAR
RESERVE 3000      ! Speicher freigeben
RESERVE 100000    ! Speicher wieder anfordern
DEFWRD "A-Z"
'
' ***** Konstanten ******
'
druckerkanal$="par:"
progtitel$="ConText-Amiga-Druck"
maxlines&=1000                    ! 1000 Zeilen Text maximal
parac$=CHR$(182)
'
' --- Attribut-Bits ---
boldb|=2
italicb|=4
underlineb|=1
subscriptb|=8
superscriptb|=16
'
' ----- DRUCKERTABELLE -----
esc$=CHR$(27)
init$=esc$+"P" ! 10 Zeichen / Zoll
reset$=CHR$(28)+CHR$(64)
fettan$=esc$+"G"
fettaus$=esc$+"H"
unterstrichenan$=esc$+"-"+CHR$(1)
unterstrichenaus$=esc$+"-"+CHR$(0)
subscriptan$=esc$+"S"+CHR$(1)
subscriptaus$=esc$+"T"
superscriptan$=esc$+"S"+CHR$(0)
superscriptaus$=esc$+"T"
kursivan$=esc$+"4"
kursivaus$=esc$+"5"
cr$=CHR$(13)  ! Wagenruecklauf
lf$=CHR$(10)  ! Linefeed
ff$=CHR$(12)  ! Seitenvorschub
linkerrand$=esc$+CHR$(108)  ! Seitenrandeinstellung
rechterrand$=esc$+"Q"+CHR$(80) ! Rechter Rand 80 vom Linken
zeilenproseite$=esc$+"C"
' ----- Uebersetzungstabelle
DIM chartab$(255)
' Amiga --> IBM-ASCII
chartab$(196)=CHR$(142) ! Ae
chartab$(214)=CHR$(153) ! Oe
chartab$(220)=CHR$(154) ! Ue
chartab$(223)=CHR$(225) ! sz
chartab$(228)=CHR$(132) ! ae
chartab$(246)=CHR$(148) ! oe
chartab$(252)=CHR$(129) ! ue
'
' ***** Variablen *****
'
DIM text$(maxlines&)
DIM attr$(maxlines&)
DIM menue$(10)
workfile$=""
bold!=FALSE
italic!=FALSE
underline!=FALSE
dwidth!=FALSE
subscript!=FALSE
superscript!=FALSE
lrand&=1
zeilenzahl&=66
texty&=1
textx&=1
lastline&=0
attribut|=0
ascii!=FALSE
contextformat!=TRUE
mitlayout!=FALSE
drucken!=FALSE
druckfertig!=FALSE
header$=""
lastprintedline&=1
'
'
'
OPENW #&H0,100,50,400,100,0,2
TITLEW #0,progtitel$
FRONTW #&H0
GOSUB printinfo
ON BREAK GOSUB ctrlc
ON ERROR GOSUB generror
' Menus initialisieren
menue$(0)="Datei"
menue$(1)="Laden"
menue$(2)="Quit"
menue$(3)=""
menue$(4)="Druck"
menue$(5)="Starten"
menue$(6)="Ende"
menue$(7)=""
menue$(8)=""
MENU menue$()
ON MENU GOSUB menureact
'
initall
DO
  SLEEP ! Menues abfragen
  GOSUB printing
  IF lastprintedline&<texty&
    LOCATE 1,7
    PRINT "Drucke Zeile ";texty&
    lastprintedline&=texty&
  ENDIF
  IF druckfertig! THEN
    PRINT
    PRINT ">> Druckende: ";workfile$
    PRINT
    PRINT ">>> Bitte Taste druecken <<<"
    WHILE INKEY$=""
    WEND
    CLS
    GOSUB printinfo
    druckfertig!=FALSE
    texty&=1
    lastprintedline&=1
    drucken!=FALSE
    GOSUB reopen
  ENDIF
LOOP
GOSUB closeall
END
'
'
> PROCEDURE initall
  OPEN "O",#7,druckerkanal$
  PRINT #7,reset$;init$;rechterrand$
RETURN
> PROCEDURE printing
  LOCAL newattr|,i&,j&
  IF lastline&<>0 AND drucken! THEN
    IF (texty&=1) AND (textx&=1) THEN
      attribut|=0
    ENDIF
    WHILE LEN(text$(texty&))=0
      PRINT #7,cr$;
      PRINT #7,lf$;
      INC texty&
      druckfertig!=texty&>lastline&
      EXIT IF texty&>lastline&
    WEND
    IF texty&<=lastline& THEN
      IF textx&>LEN(text$(texty&)) THEN
        INC texty&
        stripline
        textx&=1
        PRINT #7,cr$;
        PRINT #7,lf$;
        IF texty&>lastline& THEN
          drucken!=FALSE
          druckfertig!=TRUE
        ENDIF
      ELSE
        c$=MID$(text$(texty&),textx&,1)
        IF c$<>" " THEN
          newattr|=ASC(MID$(attr$(texty&),textx&,1))
        ENDIF
        INC textx&
        IF (newattr|<>attribut|) AND (c$<>" ") THEN
          GOSUB alleattraus
          GOSUB neueattran(newattr|)
          attribut|=newattr|
        ENDIF
        IF c$<>parac$ THEN
          IF chartab$(ASC(c$))="" THEN
            PRINT #7,c$;
          ELSE
            PRINT #7,chartab$(ASC(c$));
          ENDIF
        ENDIF
      ENDIF
    ENDIF
  ENDIF
RETURN
> PROCEDURE alleattraus
  IF bold! THEN
    bold!=FALSE
    PRINT #7,fettaus$;
  ENDIF
  IF italic! THEN
    italic!=FALSE
    PRINT #7,kursivaus$;
  ENDIF
  IF underline! THEN
    underline!=FALSE
    PRINT #7,unterstrichenaus$;
  ENDIF
```

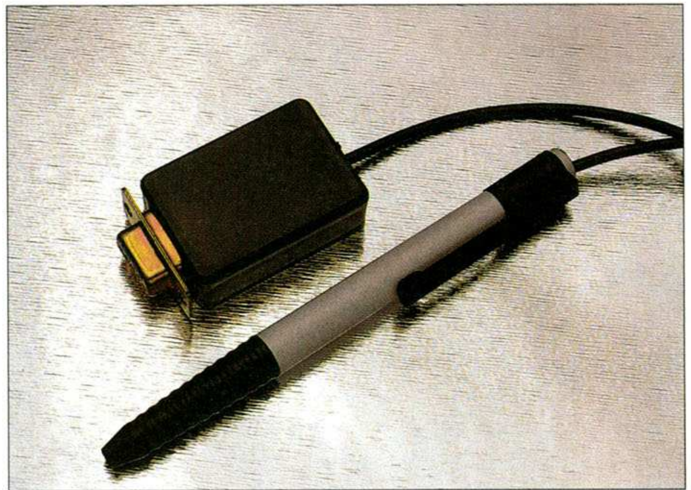
```
  IF dwidth! THEN
    dwidth!=FALSE
    ' nichts, wird nicht unterstuetzt
  ENDIF
  IF subscript! THEN
    subscript!=FALSE
    PRINT #7,subscriptaus$;
  ENDIF
  IF superscript! THEN
    superscript!=FALSE
    PRINT #7,superscriptaus$;
  ENDIF
RETURN
> PROCEDURE neueattran(attr¦)
  IF (attr¦ AND boldb¦) THEN
    bold!=TRUE
    PRINT #7,fettan$;
  ENDIF
  IF (attr¦ AND italicb¦) THEN
    italic!=TRUE
    PRINT #7,kursivan$;
  ENDIF
  IF (attr¦ AND underlineb¦) THEN
    underline!=TRUE
    PRINT #7,unterstrichenan$;
  ENDIF
  IF (attr¦ AND subscriptb¦) THEN
    subscript!=TRUE
    PRINT #7,subscriptan$;
  ENDIF
  IF (attr¦ AND superscriptb¦) THEN
    superscript!=TRUE
    PRINT #7,superscriptan$;
  ENDIF
RETURN
> PROCEDURE menureact
  LOCAL mindex&,menstr$,res&,filename$
  mindex&=MENU(0)
  menstr$=menue$(mindex&)
  IF menstr$="Quit"
    res&=1
    ALERT 0,"Druckprogramm beenden ?",2,"Na klar ! ¦ Bloss nicht !",res&
    IF res&=1 THEN
      GOSUB closeallend
    ENDIF
  ELSE IF menstr$="Laden"
    GOSUB textladen
  ELSE IF menstr$="Starten"
    IF lastline&<>0 THEN
      drucken!=TRUE
    ENDIF
  ELSE IF menstr$="Ende"
    IF drucken! THEN
      drucken!=FALSE
    ENDIF
  ENDIF
RETURN
> PROCEDURE ctrlc
  GOSUB closeallend
RETURN
> PROCEDURE generror
  GOSUB closeallend
RETURN
> PROCEDURE reopen
  PRINT #7,ff$;
  CLOSE #7
  OPEN "O",#7,druckerkanal$
  PRINT #7,reset$;init$;rechterrand$;
RETURN
> PROCEDURE closeall
  MENU KILL
  CLOSE #7
  CLOSES 1
  CLOSEW #0
RETURN
> PROCEDURE closeallend
  GOSUB closeall
  END
RETURN
> PROCEDURE textladen
  LOCAL i&,lldiv2&,dummy$
  FILESELECT "Datei Laden","Laden",workfile$,filename$
  IF filename$<>"" THEN
    somethinghappend!=FALSE
    texty&=1
    textx&=1
    OPEN "i",#1,filename$
    LINE INPUT #1,header$
    IF LEFT$(header$,7)<>"CONTEXT" THEN
      header$=""
      ascii!=TRUE
      lrand&=1
      contextformat!=FALSE
      mitlayout!=FALSE
      CLOSE #1
      OPEN "i",#1,filename$
    ELSE
      ascii!=FALSE
      IF LEN(header$)>7 THEN
        mitlayout!=TRUE
        CLOSE #1
        OPEN "i",#1,filename$
        INPUT #1,dummy$,lrand&,rrand&,zeilenzahl&
        LINE INPUT #1,dummy$
      ELSE
        contextformat!=TRUE
      ENDIF
    ENDIF
    i&=1
    WHILE NOT EOF(#1)
      LINE INPUT #1,text$(i&)
      INC i&
    WEND
    lastline&=i&-1
    IF (lastline& MOD 2<>0) AND (NOT ascii!) THEN
      wrongformaterror
    ENDIF
    lldiv2&=lastline& DIV 2
    IF NOT ascii! THEN
      FOR i&=lldiv2&+1 TO lastline&
        attr$(i&-lldiv2&)=text$(i&)
        IF LEN(text$(i&))<>LEN(text$(i&-lldiv2&)) THEN
          wrongformaterror
        ENDIF
      NEXT i&
      lastline&=lldiv2&
    ELSE
      insertattributes(0)
    ENDIF
    FOR i&=lastline&+1 TO maxlines&
      text$(i&)=""
      attr$(i&)=""
    NEXT i&
    CLOSE #1
  ENDIF
  workfile$=filename$
  PRINT #7,linkerrand$+CHR$(lrand&);
  PRINT #7,rechterrand$;
  PRINT #7,zeilenproseite$+CHR$(zeilenzahl&);
  textx&=1
  texty&=1
RETURN
> PROCEDURE insertattributes(attr¦)
  LOCAL i&,j&
  FOR i&=1 TO maxlines&
    attr$(i&)=STRING$(LEN(text$(i&)),CHR$(attr¦))
    FOR j&=1 TO LEN(text$(i&))
      IF MID$(text$(i&),j&,1)=" " THEN
        MID$(attr$(i&),j&,1)=CHR$(1)
      ENDIF
    NEXT j&
  NEXT i&
  ascii¦=FALSE
RETURN
> PROCEDURE stretchline
  IF LEN(text$(texty&))<80 THEN
    text$(texty&)=text$(texty&)+SPACE$(80-LEN(text$(texty&)))
    attr$(texty&)=attr$(texty&)+STRING$(80-LEN(attr$(texty&)),CHR$(0))
  ENDIF
RETURN
> PROCEDURE stripline
  LOCAL i&,j&,line$,comp$,ascii1$,attrstr$
  line$=text$(texty&)
  attrstr$=attr$(texty&)
  comp$=CHR$(0)
  ascii1$=CHR$(1)
  i&=LEN(line$)
  IF i&>1 THEN
    WHILE (MID$(attrstr$,i&,1)=comp$) AND (MID$(line$,i&,1)=" ")
      DEC i&
      EXIT IF i&<1
    WEND
  ENDIF
  line$=LEFT$(line$,i&)
  attrstr$=LEFT$(attrstr$,i&)
  FOR j&=i& TO 1 STEP -1
    IF (MID$(attrstr$,j&,1)=comp$) AND (MID$(line$,j&,1)=" ") THEN
      MID$(attrstr$,j&,1)=ascii1$
    ENDIF
  NEXT j&
  text$(texty&)=line$
  attr$(texty&)=attrstr$
RETURN
> PROCEDURE wrongformaterror
  LOCAL res&
  ALERT 0,"! Falsches Textformat !",2,"Weitermachen ! ¦ Vergiss es !",res&
  IF res&=2 THEN
    RUN
  ENDIF
RETURN
> PROCEDURE printinfo
  PRINT "BITTE BEACHTEN:"
  PRINT "  Auch zum Verlassen des Programms muss"
  PRINT "  der Drucker eingeschaltet und "
  PRINT "  Online sein !"
RETURN
'
'
```

ConText-Druck

Bildschirmschreiber

Der Einsatz eines Lichtgriffels am Amiga

Seit es Computer gibt, sind die Eingabegeräte wie Tastatur, Maus oder Joystick immer mehr verbessert worden. Ein Eingabegerät jedoch wird oft vergessen: der Lightpen. Die Firma Roßmöller bietet für den Amiga unter anderem auch einen LIGHTPEN, einen Lichtgriffel, an. Ob sich der Einsatz dieses Eingabegerätes lohnt, haben wir für Sie getestet.



Der Lightpen ist ein etwas ungewohntes Eingabegerät

Der LIGHTPEN wird mit einer Installationsdiskette und einer kurzen Beschreibung ausgeliefert. Nach dem Start der Diskette besteht die Möglichkeit, den LIGHTPEN auf die Koordinaten des Mauszeigers zu eichen. Dazu kann man alle vier Bewegungsrichtungen (oben, unten, rechts, links) mit den Cursortasten so lange einstellen, bis der auf den Bildschirm angesetzte LIGHTPEN genau auf dem Mauszeigerkreuz sitzt.

Danach lassen sich beliebige Disketten mit einem Steuerprogramm belegen, welches direkt nach dem Booten dieser Disketten den neuen 'Mauszeigerlenker' initialisiert.

Der LIGHTPEN im Einsatz – die Maus wird trotzdem benötigt

Beim Einsatz des LIGHTPENs am Amiga machte sich gleich der größte Nachteil bemerkbar: Die Maus muß mitbedient werden, da dem LIGHTPEN die nötigen Tasten fehlen. Somit ergibt sich beim Arbeiten mit dem LIGHTPEN folgendes Bild: In der linken Hand die Maus, in der rechten Hand den LIGHTPEN – für weitere Aktionen wird wohl kaum jemand noch eine Hand frei haben.

Sieht man von diesem Schwachpunkt einmal ab, gibt der LIGHTPEN einen guten Ersatz für die Maus ab (bis auf die fehlenden Tasten). Die Umsetzung der Koordinaten ist recht genau, und durch eine spezielle Programmierung der Treibersoftware ergibt sich eine Genauigkeit von zwei Pixel vertikal und horizontal.

Auf der Diskette zum LIGHTPEN befindet sich auch ein Testprogramm, welches die angeschlossene Hardware überprüft. Fehler können somit kaum auftauchen.

Getestet haben wir den LIGHTPEN eingehend an Deluxe Paint III von Electronic Arts. Auf dunklem Hintergrund ist aufgrund fehlender Lichtintensität keine Funktion möglich. Der Bildschirm muß möglichst hell eingestellt sein. Bei der Freihandfunktion ist eine sehr ruhige Hand vonnöten, um einigermaßen saubere Zeichnungen zu bekommen. Positionieren ließ sich der LIGHTPEN gut, Zittern und ungewolltes Wechseln der Position verschwanden nach Einstellung des Kontrastes und der Helligkeit am Monitor.

Maus und LIGHTPEN – ein unzertrennliches Gespann?

Der LIGHTPEN lohnt sich dann, wenn man wenig Platz für Mausbewegungen hat. Ein großes Manko ist und bleibt jedoch die weitere Bedienung der Maus. Das Fazit kann daher nur lauten: für ernsthafte Anwendungen kaum geeignet, für eigene Versuche ein brauchbares Objekt.

(jb)

AMIGA DOS Blitzlicht

Name: LIGHTPEN

Vertrieb: Roßmöller Handshake, Neuer Markt 21 D-5309 Meckenheim Tel.:02225/2061-62

Preis: 79,95 DM
Zubehör: LJ-Adapter zum gleichzeitigen Anschluß der Joysticks und des LIGHTPENs (29,95 DM)

Positiv:
- leichte Installation
- gute Handhabung
- läuft unter vielen Programmen

Negativ:
- Maustasten müssen separat betrieben werden
- nur bei sehr hellem Hintergrund ist eine ruhige Zeichnung möglich
- relativ hoher Preis

Brennpunkt Amiga

Der EPROM-Brenner VESUV von Roßmöller

Stellen Sie sich ein fast unscheinbares, schwarzes Kästchen vor, auf der einen Seite mit einer Centronics-(Drucker-) Buchse, auf der anderen mit einem C-64-Userport-Anschluß bestückt, mit einem seltsamen Sockel für integrierte Schaltungen und einer Leuchtdiode auf der Oberseite sowie einer Anzahl beiliegender Disketten. Was das sein soll? Der VESUV-EPROM-Brenner von der Firma Roßmöller.

EPROMs haben bekanntlich die gute Eigenschaft, eingebrannte (durch Anlegung einer bestimmten Spannung wissentlich herbeigeführte Zerstörungen in der Kristallstruktur eines Halbleiters) Daten auf längere Zeit zu behalten. Einsetzbar sind diese EPROM-Bausteine dann als Ersatz für im Rechner vorhandene ROM-Bausteine; ein Vorteil, wenn man Teile eines Betriebssystems seinen eigenen Anforderungen anpassen will.

Der VESUV der Firma Roßmöller Handshake unterscheidet sich von anderen EPROM-Brennern dadurch, daß hier die übertragenen Daten nicht über einen direkten Prozessorport (Erweiterungsport) an den Brenner gelangen, sondern von der Druckerschnittstelle aus. Der Vorteil liegt klar auf der Hand: Der VESUV ist für viele Computertypen gedacht, einzig die Steuersoftware muß an die unterschiedlichen Betriebssysteme angepaßt werden. Und so befindet sich im Lieferumfang des VESUV je eine Softwarediskette für Amiga und Atari im 3,5-Zoll-Format und je eine 5,25-Zoll-Diskette für C-64 und IBM-PC und Kompatible.

Da der C64 nicht über einen Parallelport (Druckeranschluß) verfügt, wurde dem VESUV noch ein Userport-Anschluß eingebaut, damit auch Besit-

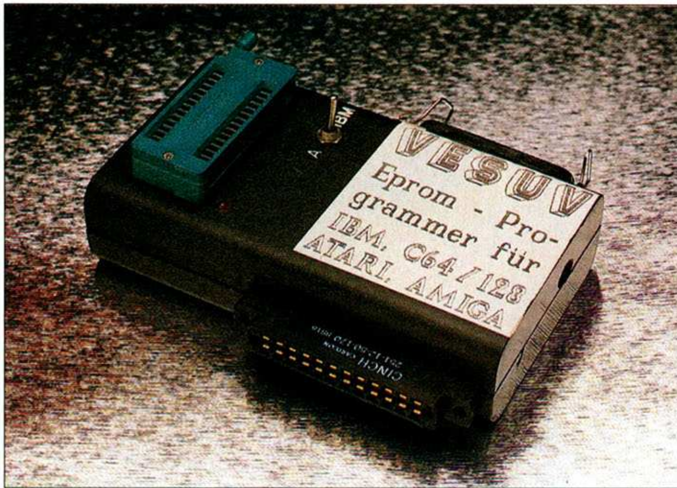


Bild 1: Der VESUV von Roßmöller brennt EPROMs der Typen 27XX und 25XX



Bild 2: Das Menü der Amiga-Software ist übersichtlich und schnell verständlich

zer von Commodores 8-Bitter den Brenner nutzen können.

Die Anleitung ist zwar etwas kurz, reicht jedoch für den Einsatz des Brenners am Amiga vollkommen aus, vor allem, wenn man sich an dieselbe hält.

Der VESUV benötigt zum Betrieb noch ein eigenes Netzteil (außer beim C-64, hier wird die Versorgungsspannung vom Rechner geliefert). Für Bastler sei gesagt, daß dieses Netzteil eine Spannung von 7,5 bis 9 Volt erzeugt sowie für eine Stromstärke von mindestens 250 Milliampere ausgelegt sein muß. Für Nichtelektroniker bietet die Firma Roßmöller ein separates Netzteil für knapp 20 DM an.

Auf die Einhaltung der Strom- und Spannungswerte ist aber auf jeden Fall zu achten, da aus der Netzteilspannung die EPROM-Brennspannungen erzeugt werden. Und deshalb sei den "Selbstbauern" geraten, gerade beim Netzteil Sorgfalt walten zu lassen.

Gut gebrannt hält länger – dies gilt vor allem für EPROMs

Nach dem Start der Software erwartet den zukünftigen 'Chipbrenner' ein grafisch sehr ansprechendes Menü mit folgenden Funktionen:

- EPROM brennen
- EPROM lesen
- EPROM Verify
- Leertest
- Monitor
- EPROM Programmtest
- DISK I/O
- ENDE

Zusätzlich sind alle brennbaren EPROM-Typen gekennzeichnet. Die Auswahl geschieht dadurch, daß man eines der stilisierten EPROMs mit der Maus anklickt und in den ebenfalls gezeichneten Sockel setzt. Dies sollte man natürlich auch mit dem Original tun. Danach kann das EPROM ausgelesen, beschrieben oder getestet werden. Ein integrierter Speichermonitor läßt Manipulationen an Bits und Bytes zu, so daß sich jeder sein EPROM-Wunschprogramm zusammenstricken kann.

Gebrannt werden können EPROMs der Typenreihe 27XX – angefangen beim 2708 bis hin zum 27011. Darunter befindet sich auch der Typ 2508 mit einfacher Stromversorgung.

Angezeigt werden neben den Menüfunktionen und den EPROM-spezifischen Werten noch die erzeugte Programmierspannung, die bei den einzelnen Typen variiert, sowie der Programmiermodus.

Ebenfalls zu sehen sind die Anfangs- und die Endadresse des benutzten Speicherbereiches sowie die Anfangsadresse des EPROMs.

Mit der Funktion DISK I/O lassen sich Programme von Diskette in den Speicherbereich laden, dort verändern und anschließend in ein EPROM brennen. EPROM VERIFY und EPROM PRGTEST dienen anschließend dem Testen des Bausteines.

Zusätzlich zum Hauptprogramm befinden sich auf der Amiga-Diskette noch mehrere Info-Files, die weiteren Aufschluß über die Bedienung des EPROM geben, sowie eine Anleitung zum Erzeugen eines neuen Kickstart-ROMs. Kickstart läßt sich durch seine Größe nicht in ein EPROM brennen. Es werden hierfür vier Stück des Typs 27256 mit je 64 kByte Speicherplatz benötigt. Das Kickstart-ROM kann ausgelesen und auf Diskette gespeichert werden, wenn es sich um einen A 500 oder A 2000 handelt. Beim A 1000 reicht das Einlegen der Kickstart- Diskette.

Diese wird eingelesen, die enthaltenen Daten so geteilt, daß sie brennfähig im Speicher stehen. Die Firma Roßmöller bietet zu diesem Zweck eine dreifache Umschaltplatine für verschiedene Kickstart-Versionen an, auf der einmal das Original-Kickstart-ROM (falls vorhanden), ein zweites ROM und die vier EPROMs sitzen können.

Der VESUV ist ein rundherum gelungenes Gerät

Wer einen EPROM-Brenner benötigt, sollte am VESUV nicht vorbeigehen. Schon allein die Möglichkeit, das Gerät an verschiedenen Rechnern einzusetzen, ohne irgendein Zusatzteil (bis auf das Netzteil, aber das dürfte wohl kein Problem sein) zu benötigen, weil die komplette Software-Reihe für Amiga, Atari, PC und C-64 schon beiliegt, hebt den VESUV von anderen EPROM-Brennern ab. Da ist auch der Preis von 249,– DM nicht zu hoch angesetzt. Der einzige Schwachpunkt, der uns auffiel, ist, daß sämtliche Softwarefunktionen abliefen, obwohl kein Brenner angeschlossen war. Dies ist jedoch eher eine Gewöhnungssache als ein Manko, denn erstens wird man es schnell merken, und zweitens tritt nirgendwo ein Defekt deswegen auf.

Zu den positiven Punkten ist vor allem die klare Benutzerführung zu nennen, es macht richtig Spaß, mit der Maus eines der gezeichneten EPROMs in den ebenfalls gezeichneten Sockel zu setzen. Durch die leichte Bedienung wird es auch Einsteigern erleichtert, sich mit EPROMs zu beschäftigen.

Deswegen kann man dem VESUV ruhigen Gewissens die Note 'sehr gut' geben.

(jb)

AMIGA DOS Blitzlicht

Name: VESUV-EPROM-Brenner
Vertrieb: Roßmöller Handshake, Neuer Markt 21 D-5309 Meckenheim Tel.:02225/2061-62
Preis: 249,– DM
Zubehör: Netzteil für 19,95 DM

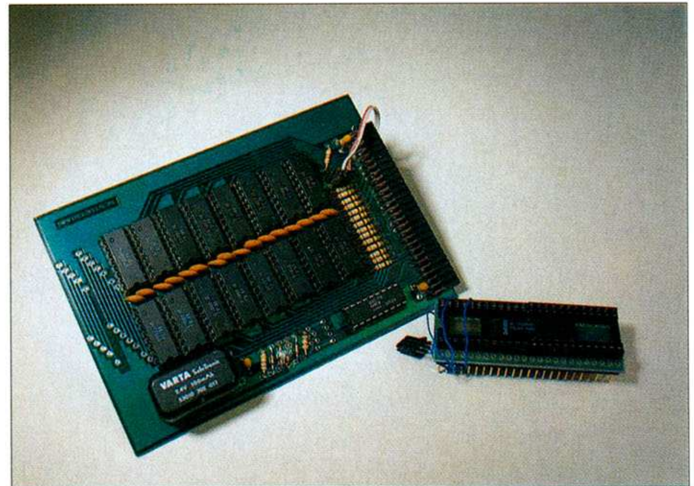
Positiv:
- über Druckerport anschließbar
- Software für mehrere Computertypen
- gute, übersichtliche Menüführung
- Kickstart-EPROM-Erstellung automatisiert
- inklusive Testprogramm für Hardware
- klein und platzsparend

Negativ:
- Funktionen laufen auch ohne Brenner
- Kontrolle für Spannungsversorgung fehlt

Speicher für den Kleinsten

Die 2-MByte-Speichererweiterung für den Amiga 500

Der Kleinste der Amigas, der A 500, ist in seiner Grundkonfiguration mit 512 kByte und einem Laufwerk nur sehr eingeschränkt nutzbar, viele Anwendungsprogramme benötigen sehr viel mehr Speicher. Leider sind Speichererweiterungen nicht gerade billig, vor allem wegen der Preise für die RAM-Bausteine. So steht man vor der Wahl: Laufwerk oder Speichererweiterung?



Die Roßmöller-Speichererweiterung läßt sich sehr schnell und problemlos in den Amiga 500 einbauen

Zuerst wird daher meistens Laufwerk Nummer 2 gekauft, dieses benötigt jedoch ebenfalls einen Teil des Arbeitsspeichers, so daß sich dieser wiederum vermindert. Speichererweiterungen mit nochmals 512 kByte gibt es ja inzwischen wie Sand am Meer, aber warum soll es nicht gleich ein bißchen mehr sein?

A2MB – Speichergigant zu niedrigem Preis mit hoher Leistung

Die Speichererweiterung A2MB von der Firma Roßmöller Handshake hat genau das richtige Format, um den A 500 aufzuwerten. Satte 2 MByte Speicher sowie eine akkugepufferte Echtzeituhr befinden sich auf einem 127 x 90 mm großen Platinchen, das mit diesen Maßen Aufnahme in den Speichererweiterungsport an der Unterseite des Amigas findet. Um diesen Speicher nutzen zu können, muß allerdings noch GARY, einer der im A 500 enthaltenen Customchips aus seinem Sockel gehebelt werden, da für die neue Speichermenge eine neue Ansteuerlogik erforderlich ist. Diese liegt der RAM-Karte als Sockel mit integrierter Schaltung bei, der in den nun leeren GARY-Sockel gesteckt wird und in dem wiederum Gary seinen Platz findet.

Die Speichererweiterung wird nun noch mit einem Anschlußdraht mit dem GARY-Sockel verbunden. Damit wäre der Einbau erledigt.

Von der RAM-Karte führt außerdem noch ein Schalter weg, den man sich irgendwo an einer zugänglichen Stelle des Gehäuses einbauen sollte. Er schaltet die Speichererweiterung an und aus. Dieses ist bei einigen Programmen, vornehmlich Spielen, noch immer vonnöten, denn der Aus- und Wiedereinbau einer RAM-Karte ist nicht unbedingt jedesmal zu empfehlen.

Mehr Speicher im Amiga 500 – ein helles Licht am Horizont

Der Preis für diese Speichererweiterung wurde uns mit 648,– DM angegeben, jedoch variieren die Preise für RAM-Bausteine derart, daß man den aktuellen Preis lieber telefonisch erfragen sollte. Dank stromsparender 1-MByte-Chips und dank des relativ niedrigen Preises (der sich, wie gesagt, ändern kann), erhält der A-500-Besitzer ein sehr gutes Zusatzgerät für seinen Computer, das den Amiga gleichzeitig zu einer voll aufgerüsteten Maschine macht.

(jb)

AMIGA DOS Blitzlicht

Name: A2MB-Speichererweiterung
Anbieter: Roßmöller Handshake, Neuer Markt 21
D-5309 Meckenheim
Tel.:02225/2061-62
Preis: 648,– DM

Positiv:
- Aufnahme im Speichererweiterungsschacht
- schneller Einbau ohne Komplikationen
- eingebaute Echtzeituhr
- abschaltbar

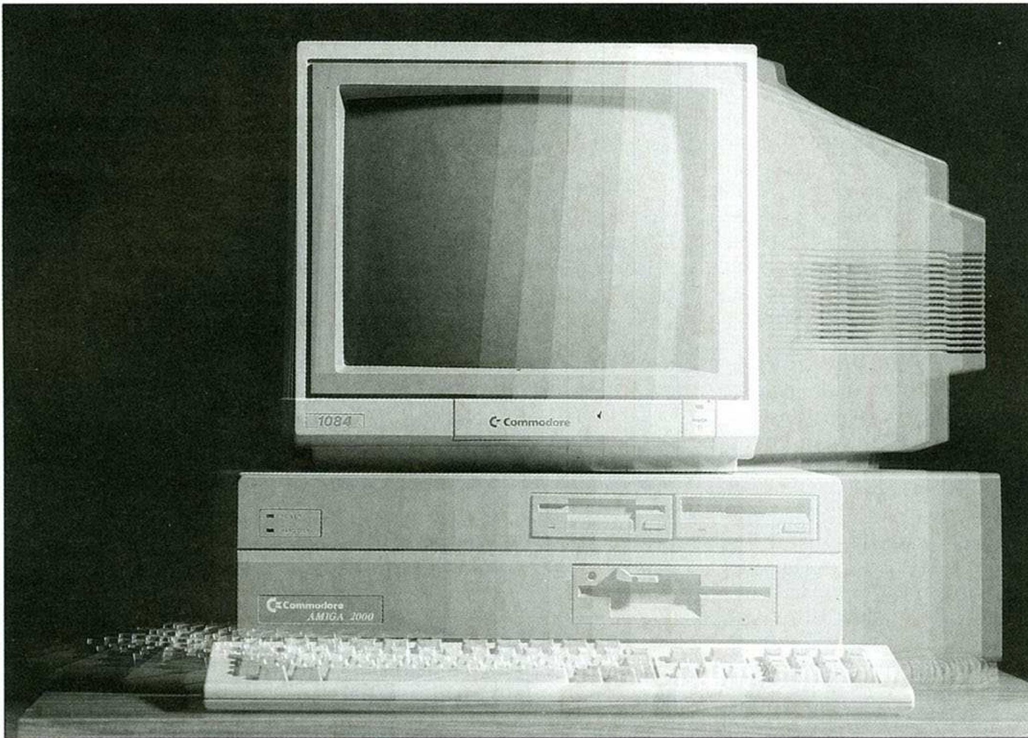
Negativ:
- keine Negativpunkte zu finden

Das ist Ihre Chance...

Schon eine Kleinanzeige bringt oftmals großen Erfolg und hilft, neue Kontakte zu knüpfen.
Nutzen Sie unser Angebot und profitieren Sie von der Tatsache, daß unsere Zeitschrift

»AMIGA DOS«

jeden Monat von zigtausend Computer-Interessierten gelesen wird.
Möchten Sie etwas verkaufen, tauschen, oder suchen Sie das »Tüpfelchen auf dem i« Ihre Annonce erscheint dann in der nächsterreichbaren Ausgabe.
Wir möchten ausdrücklich darauf hinweisen, daß wir keine Anzeigen veröffentlichen, aus denen ersichtlich ist, daß es sich hierbei um Veräußerungen von Raubkopien handelt.
Des weiteren machen wir darauf aufmerksam, daß indizierte Computerspiele nicht in Form von Anzeigen beworben werden dürfen.



Achim Stindt

Rundherum, das ist nicht schwer...

Rotationskörper leicht erstellt

Auf Anhieb fallen mir eigentlich nur zwei große Anwendungsmöglichkeiten für den Computer ein, deren Vorteile für einen Laien deutlich sichtbar sind. Zum einen sind es riesige Roboterstraßen, die ohne irgendein menschliches Zutun ein halbes oder auch ganzes Auto zusammensetzen, zum anderen sind es Filme aus dem Computer, die nicht einen einzigen Schauspieler benötigen.

Nun, ein derartiger Leistungsbeweis darf in diesem Heft natürlich nicht fehlen. Einen kleinen Roboter kann man natürlich auch mit dem Amiga steuern, doch erfordert dies einiges an bastlerischem Können, vor allem aber Zeit und Geld für die zusätzlichen Geräte. Deshalb hier ein Beispiel für gekonnte Computergrafik, die unbestritten ein Spezialgebiet des Amiga ist, und dort wird er ja auch oft professionell eingesetzt. Dieses Programm ist natürlich schon eine Nummer kleiner, aber eine gewisse Faszination möchte ihm wohl niemand absprechen. Doch was macht es eigentlich? Man denke sich einen beliebigen Körper im dreidimensionalen Raum. Dieser kann derartig nicht von einem Computer dargestellt werden, er muß erst einer Dimension "beraubt" werden, um ihn auf dem ebenen Bildschirm darzustellen.

Nichts anderes macht das Programm. Es ist schon schwierig, einen Körper auf dem Bildschirm zu definieren, eben weil man ihn nicht einfach darstellen kann. Deshalb greift dieses Programm auf einen Spezialfall eines Körpers zurück: den sogenannten Rotationskörper.

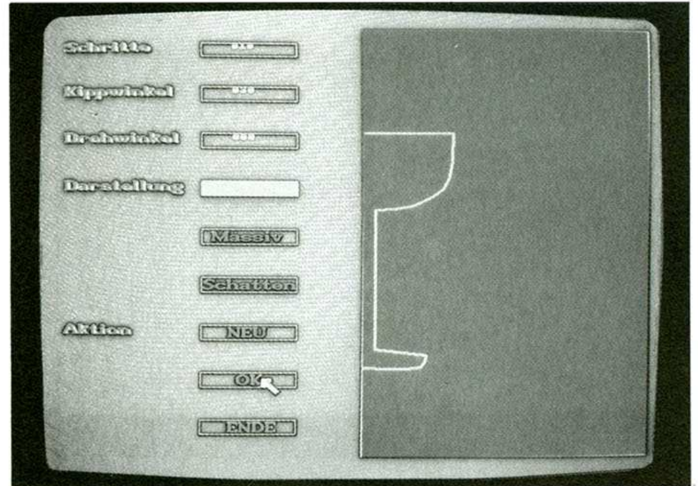
Rotierende Körper – oder die Schwierigkeit, aus einem 2D ein 3D zu machen

Der Rotationskörper ist zum Beispiel ein rundes Glas oder eine Flasche. Man nimmt eine Außenlinie und läßt diese um eine Achse im Raum rotieren. Diese Außenlinie kann man in der rechten Bildschirmhälfte festlegen: mit der Maus auf einen gewünschten Punkt zeigen und dann die linke Maustaste drücken. Die so angewählten Punkte werden verbunden und ergeben die Außenlinie. Die Mittelachse, um die später rotiert wird, ist dabei die Senkrechte in der Mitte des Bildschirms. Wählt man dann die Funktion [OK] an, so wird der Rotationskörper berechnet und dargestellt. Dabei kann man eine von drei Darstellungsarten wählen: Das Gadget [Draht] bedeutet, daß der Körper als Drahtgittermodell dargestellt wird. Wenn das [Massiv]-Gadget gewählt ist, wird der Körper

aus Flächen zusammengesetzt. Der ganz besondere Effekt ist aber die Simulation einer Lichtquelle: Dabei werden für die verschiedenen Flächen auch verschiedene Farben errechnet. Dieser Effekt wird durch das Gadget [Schatten] ausgewählt. Weiterhin ist bei dieser Berechnung die Möglichkeit gegeben, den Betrachtungspunkt frei zu ändern: Mit den Parametern [Drehwinkel] und [Kippwinkel] kann man angeben, wie weit die Blickrichtung von der Waagerechten abweicht und wie stark der Körper gekippt ist. Der letzte Parameter [Schritte] gibt an, wie viele Schritte bei der Rotation gemacht werden. Je mehr Schritte berechnet werden sollen, desto länger rechnet der Computer, desto runder erscheint aber auch das Bild.

Das Programm ist in Modula-2 (A&L Compiler) verwirklicht. Dies bedeutet eine hohe Geschwindigkeit und eine gute Ausnutzung der Amiga-Betriebssystemroutinen, ganz abgesehen von der guten Lesbarkeit des Programms. Programmiertechnisch gesehen ist besonders die Anmeldung der Gadgets interessant. Da das Betriebssystem die Möglichkeiten so richtig universell gelassen haben, brauchen alle Routinen auch sehr viele Angaben. Um einen eigenen Bildschirm zu öffnen, benötigt man tatsächlich schon zirka 50 Zeilen. Eine einzelnes Gadget anzumelden benötigt eigentlich knapp weniger Platz. Da lohnt es sich dann, Möglichkeiten zu überlegen, um Platz zu sparen. Die folgende Möglichkeit fand dabei Verwendung: Da viele Angaben der verschiedenen Gadgets identisch sind, wird eine Routine geschrieben, die ein Gadget prinzipiell vorbereitet und dann nur noch die spezifischen Parameter hinzufügt. In Modula finden die Gadgets dann auch in einem Feld (!) Platz. Das Programm benutzt den Hires-Interlace Bildschirm mit einer Auflösung von 640x512 Punkten. Auf diesem Bildschirm wirkt der original Zeichensatz ziemlich verloren, deshalb wird ein größerer benutzt, der allerdings von Diskette nachgeladen wird. Der Font "Ruby 15" muß sich also auf der Diskette, mit der der Rechner gebootet wurde, in den entsprechenden Direktories befinden (dies trifft bei der Original-Workbench von Commodore zu).

Somit bleibt uns nur noch, Ihnen viel Spaß mit dem Programm zu wünschen, und daß alle Ihre selbsterstellten Rotationskörper fest auf der Achse bleiben. (jb)



Aus ein paar Strichen, gezogen mit der Maus, werden nach kurzer Berechnungszeit komplexe Figuren. Einfacher kann die Gestaltung von Rotationskörpern wohl kaum vonstatten gehen

Listings



```
*****************************************************************
*                      AMIGA DOS INFO                           *
*    Programmname    : ROD                                      *
*    Funktion        : Erstellt Rotationskorper                 *
*    Sprache         : MODULA 2                                 *
*    Besonderheiten  : Drei Darstellungsmoglichkeiten           *
*****************************************************************

(* ###################################
   #                                 #
   #       -----------------         #
   #        Rotationskörper          #
   #       -----------------         #
   #                                 #
   #           (C) 1989              #
   #                                 #
   #  Achim Stindt & Ingmar Reyer    #
   #                                 #
   #         und AMIGA DOS           #
   #                                 #
   ################################### *)


MODULE Rod ;


  FROM Exec          IMPORT UByte, GetMsg, ReplyMsg, WaitPort ;
  FROM Heap          IMPORT AllocMem, Deallocate, Largest ;
  FROM SYSTEM        IMPORT ADR, ADDRESS, LONGSET ;
  FROM Strings       IMPORT Length, Insert ;
  FROM Graphics      IMPORT ViewPortPtr, RastPortPtr, WritePixel, Draw, RectFill,
                            ClearScreen, ViewModes, ViewModeSet, DrawModeSet,
                            RastPortFlags, RastPortFlagSet, SetAPen, Move, jam1,
                            AreaInfoPtr, TmpRasPtr, InitArea, InitTmpRas,
                            AreaMove, AreaDraw, AreaEnd, Text, FontStyles,
                            FontFlagSet, FontStyleSet, TextAttr, TextAttrPtr,
                            FontFlags, TextFontPtr, SetFont, CloseFont,
                            DrawModes, SetDrMd, LoadRGB4 ;
  FROM MathLib0      IMPORT sqrt, arctan, sin, cos ;
  FROM DiskFont      IMPORT OpenDiskFont ;
  FROM Intuition     IMPORT NewWindow, IDCMPFlagSet, IDCMPFlags, ScreenFlags,
                            ScreenFlagSet, WindowPtr, WindowFlags, WindowFlagSet,
                            OpenWindow, CloseWindow, NewScreen, ScreenPtr,
                            OpenScreen, CloseScreen, customScreen, gadgHNone,
                            IntuiText, Gadget, boolGadget, Border, GadgetPtr,
                            GadgetFlags, GadgetFlagSet, ActivationFlags,
                            ActivationFlagSet, AddGadget, RemoveGadget,
                            RefreshGadgets, strGadget, StringInfo,
                            IntuiMessage, IntuiMessagePtr, DisplayBeep ;
  FROM Conversions IMPORT StrToVal ;

  CONST
    MyTextName = "Ruby.font" ;

  TYPE
    Vector = RECORD x,y,z : REAL ; END;
    Plain = RECORD n : Vector ; k : REAL ; END ;
    gestalttype = ( draht, filled, schatten ) ;

  VAR
    MyWindowPtr : WindowPtr ;
    MyNewScreen : NewScreen ;
    MyNewWindow : NewWindow ;
    MyScreenPtr : ScreenPtr ;
    MyRastPortPtr : RastPortPtr ;
    MyViewPortPtr : ViewPortPtr ;
    MyTmpRasPtr : TmpRasPtr ;
    MyAreaInfoPtr : AreaInfoPtr ;
    MytextAttr : TextAttr ;
    MyTextAttrPtr : TextAttrPtr ;
    MyTextFontPtr : TextFontPtr ;
    MyIntuiMessagePtr : IntuiMessagePtr ;
    MyIntuiMessage : IntuiMessage ;
    MyClass : IDCMPFlagSet ;
    MyBufferPtr : ADDRESS ;
    MyIntuiText : ARRAY [ 0 .. 9 ] OF IntuiText ;
    MyGadgetPtr : GadgetPtr ;
    MyGadget : ARRAY [ 0 .. 9 ] OF Gadget ;
    MyStringInfo : ARRAY [ 0 .. 2 ] OF StringInfo ;
    Mybuffer : ARRAY [ 0 .. 2 ] OF ARRAY [ 0 .. 3 ] OF CHAR ;
    MyColor : ARRAY [ 0 .. 15 ] OF INTEGER ;
    MyRahmen : Border ;
    xyr : ARRAY [ 0 .. 9 ] OF INTEGER ;
    ZentrumPunkt, ObjectPunkt, Normale1, Normale2,BildPunkt, AbstandPunkt,
    Schnittpunkt, LichtPunkt : Vector ;
    gestalt : gestalttype ;
    BildEbene, AbstandEbene1, AbstandEbene2, ZentrumEbene : Plain ;
```

```
    i, j, l, step, dint, ax, ay, ex, ey : INTEGER ;
    max, may, dlong : LONGINT ; frischline : BOOLEAN ;
    O, RO, RS : ARRAY [ 0 .. 63 ] OF ARRAY [ 0 .. 1 ] OF INTEGER ;
    C : ARRAY [ 0 .. 63 ] OF ARRAY [ 0 .. 13 ] OF INTEGER ;
    LeftButton [ 0BFE001H ] : SET OF [ 0 .. 7 ] ;

(******************************************************************************
 *                                                                            *
 * Screen                                                                     *
 *                                                                            *
 ******************************************************************************)

  PROCEDURE OpenDisplay () : BOOLEAN ;
  BEGIN
    WITH MyNewScreen DO
      leftEdge := 0 ;
      topEdge := 0 ;
      width := 640 ;
      height := 512 ;
      depth := 4 ;
      detailPen := 0 ;
      blockPen := 1 ;
      viewModes := ViewModeSet { hires, lace } ;
      type := customScreen ;
      font := NIL ;
      defaultTitle := NIL ;
      gadgets := NIL ;
      customBitMap := NIL ;
    END ;
    MyScreenPtr := OpenScreen ( MyNewScreen ) ;
    IF MyScreenPtr = NIL THEN RETURN FALSE ; END ;
    MyViewPortPtr := ADR ( MyScreenPtr^.viewPort ) ;
    WITH MyNewWindow DO
      leftEdge := 0 ;
      topEdge := 0 ;
      width := 640 ;
      height := 512 ;
      detailPen := 0 ;
      blockPen := 1 ;
      idcmpFlags := IDCMPFlagSet { gadgetDown, mouseMove } ;
      flags := WindowFlagSet { reportMouse, activate,
        noCareRefresh, borderless, rmbTrap } ;
      title := NIL ;
      type := customScreen ;
      firstGadget := NIL ;
      checkMark := NIL ;
      screen := MyScreenPtr ;
      bitMap := NIL ;
      minWidth := 0 ;
      minHeight := 0 ;
      maxWidth := 640 ;
      maxHeight := 512 ;
    END ;
    MyWindowPtr := OpenWindow ( MyNewWindow ) ;
    IF MyWindowPtr = NIL THEN CloseScreen ( MyScreenPtr ) ; RETURN FALSE ; END ;
    MyRastPortPtr := MyWindowPtr^.rPort ;
    WITH MytextAttr DO
      name := ADR ( MyTextName ) ;
      ySize := 15 ;
      style := FontStyleSet { } ;
      flags := FontFlagSet { diskFont } ;
    END ;
    MyTextAttrPtr := ADR ( MytextAttr ) ;
    MyTextFontPtr := OpenDiskFont ( MyTextAttrPtr ) ;
    IF MyTextFontPtr = NIL THEN CloseWindow ( MyWindowPtr ) ; CloseScreen ( MyScreenPtr ) ; RETURN FALSE ; END ;
    SetFont ( MyRastPortPtr, MyTextFontPtr ) ;
    IF Largest ( TRUE ) < 10400 THEN CloseWindow ( MyWindowPtr ) ; CloseScreen ( MyScreenPtr ) ; RETURN FALSE ; END ;
    AllocMem ( MyBufferPtr, 42000, TRUE ) ;
    MyTmpRasPtr := MyBufferPtr ;
    INC ( MyBufferPtr, 8 ) ;
    InitTmpRas ( MyTmpRasPtr^, MyBufferPtr, 40960 ) ;
    MyRastPortPtr^.tmpRas := MyTmpRasPtr ;
    INC ( MyBufferPtr, 40960 ) ;
    MyAreaInfoPtr := MyBufferPtr ;
    INC ( MyBufferPtr, 28 ) ;
    InitArea ( MyAreaInfoPtr^, MyBufferPtr, 5 ) ;
    MyRastPortPtr^.areaInfo := MyAreaInfoPtr ;
    DEC ( MyBufferPtr, 40996 ) ;
    MyRastPortPtr^.aOlPen := 3 ;
    MyRastPortPtr^.drawMode := jam1 ;
    MyColor [  0 ] := 0040H ;
    MyColor [  1 ] := 0292H ;
    MyColor [  2 ] := 0B05H ;
    MyColor [  3 ] := 0CA0H ;
    MyColor [  4 ] := 0444H ;
    MyColor [  5 ] := 0555H ;
    MyColor [  6 ] := 0666H ;
    MyColor [  7 ] := 0777H ;
    MyColor [  8 ] := 0888H ;
    MyColor [  9 ] := 0999H ;
    MyColor [ 10 ] := 0AAAH ;
    MyColor [ 11 ] := 0BBBH ;
    MyColor [ 12 ] := 0CCCH ;
    MyColor [ 13 ] := 0DDDH ;
    MyColor [ 14 ] := 0EEEH ;
    MyColor [ 15 ] := 0FFFH ;
    LoadRGB4 ( MyViewPortPtr, ADR ( MyColor [ 0 ] ), 16 ) ;
    RETURN TRUE ;
  END OpenDisplay ;

  PROCEDURE CloseDisplay ;
  BEGIN
    Deallocate ( MyBufferPtr ) ;
    CloseFont ( MyTextFontPtr ) ;
    CloseWindow ( MyWindowPtr ) ;
    CloseScreen ( MyScreenPtr ) ;
  END CloseDisplay ;

(******************************************************************************
 *                                                                            *
 * Gadgets                                                                    *
 *                                                                            *
 ******************************************************************************)

  PROCEDURE MakeGadget ( y, ID : INTEGER ; text : ARRAY OF CHAR ; art : CARDINAL ) ;
    VAR
      dbool : BOOLEAN ; dtext : ARRAY [ 0 .. 8 ] OF CHAR ;
  BEGIN
    WITH MyGadget [ ID ] DO
      leftEdge := 150 ;
      topEdge := y ;
      width := 100 ;
      height := 16 ;
      flags := gadgHNone ;
      activation := ActivationFlagSet { relVerify, gadgImmediate } ;
      gadgetType := boolGadget ;
      gadgetRender := ADR ( MyRahmen ) ;
      selectRender := NIL ;
      gadgetText := ADR ( MyIntuiText [ ID ] ) ;
      mutualExclude := LONGSET { 0 } ;
      specialInfo := NIL ;
      gadgetID := ID ;
      userData := NIL ;
      nextGadget := NIL ;
      CASE art OF
      2 :
        flags := GadgetFlagSet { } ;
        activation := activation +
          ActivationFlagSet { toggleSelect, longint, stringCenter } ;
        gadgetType := strGadget ;
        gadgetText := NIL ;
        specialInfo := ADR ( MyStringInfo [ ID ] ) ;
        WITH MyStringInfo [ ID ] DO
          buffer := ADR ( Mybuffer [ ID ] ) ;
          undoBuffer := buffer ;
          maxChars := 4 ;
          dispPos := 0 ;
          bufferPos := 0 ;
          StrToVal ( Mybuffer [ ID ], dlong, dbool, 10, dbool ) ;
          longInt := dlong ;
        END ;
      |
      3 :
        leftEdge := 320 ;
        topEdge := 0 ;
        width := 320 ;
        height := 512 ;
        gadgetRender := NIL ;
        gadgetText := NIL ;
      |
      ELSE
      END ;
    END ;
    WITH MyIntuiText [ ID ] DO
      frontPen := 3 ;
      backPen := 1 ;
      drawMode := jam1 ;
      leftEdge := 1 ;
      topEdge := 2 ;
      iTextFont := MyTextAttrPtr ;
      nextText := NIL ;
      iText := ADR ( text ) ;
    END ;
    WITH MyRahmen DO
      leftEdge := -1 ;
      topEdge := -1 ;
      frontPen := 2 ;
      backPen := 0 ;
      drawMode := jam1 ;
      count := 5 ;
      xy := ADR ( xyr ) ;
      nextBorder := NIL ;
    END ;
    xyr [ 0 ] :=   0 ;
    xyr [ 1 ] :=   0 ;
    xyr [ 2 ] := 101 ;
    xyr [ 3 ] :=   0 ;
    xyr [ 4 ] := 101 ;
    xyr [ 5 ] :=  17 ;
```

Rotationskörper

```
    xyr [ 6 ] :=  0 ;
    xyr [ 7 ] := 17 ;
    xyr [ 8 ] :=  0 ;
    xyr [ 9 ] :=  0 ;
    MyGadgetPtr := ADR ( MyGadget [ ID ] ) ;
    dlong := AddGadget ( MyWindowPtr, MyGadgetPtr, -1 ) ;
    RefreshGadgets ( MyGadgetPtr, MyWindowPtr, NIL ) ;
  END MakeGadget ;

  PROCEDURE Turn ( ID : gestalttype ) ;
    VAR
      h : INTEGER ;
  BEGIN
    h := 188 + 56 * ORD ( ID ) ;
    SetDrMd ( MyRastPortPtr, DrawModeSet { complement } ) ;
    RectFill ( MyRastPortPtr, 150, h, 249, h + 15 ) ;
    SetDrMd ( MyRastPortPtr, jam1 ) ;
  END Turn ;

  PROCEDURE Select ( ID : gestalttype ) ;
  BEGIN
    Turn ( gestalt ) ;
    gestalt := ID ;
    Turn ( gestalt ) ;
  END Select ;

(*******************************************************************************
*                                                                             *
* The Soft Bulldog                                                            *
*                                                                             *
*******************************************************************************)

  PROCEDURE Quad ( x1, y1, x2, y2, x3, y3, x4, y4 : INTEGER ) ;
  BEGIN
    IF gestalt # draht THEN
      MyRastPortPtr^.flags := RastPortFlagSet { areaOutline } ;
      dlong := AreaMove ( MyRastPortPtr, x1, y1 ) ;
      dlong := AreaDraw ( MyRastPortPtr, x2, y2 ) ;
      dlong := AreaDraw ( MyRastPortPtr, x3, y3 ) ;
      dlong := AreaDraw ( MyRastPortPtr, x4, y4 ) ;
      dlong := AreaEnd ( MyRastPortPtr ) ;
      MyRastPortPtr^.flags := RastPortFlagSet { } ;
    ELSE
      Move ( MyRastPortPtr, x1, y1 ) ;
      Draw ( MyRastPortPtr, x2, y2 ) ;
      Draw ( MyRastPortPtr, x3, y3 ) ;
      Draw ( MyRastPortPtr, x4, y4 ) ;
      Draw ( MyRastPortPtr, x1, y1 ) ;
    END ;
  END Quad ;

  PROCEDURE PrintAt ( x, y : INTEGER ; text : ARRAY OF CHAR ) ;
    VAR
      dtext : ARRAY [ 0 .. 39 ] OF CHAR ;
      fgPn : INTEGER ;
  BEGIN
    fgPn := INTEGER ( MyRastPortPtr^.fgPen ) ;
    dtext := "" ;
    Insert ( dtext, 0, text ) ;
    SetAPen ( MyRastPortPtr, 2 ) ;
    Move ( MyRastPortPtr, x + 2, y + 1 ) ;
    Text ( MyRastPortPtr, ADR ( dtext ), Length ( dtext ) ) ;
    SetAPen ( MyRastPortPtr, fgPn ) ;
    Move ( MyRastPortPtr, x, y ) ;
    Text ( MyRastPortPtr, ADR ( dtext ), Length ( dtext ) ) ;
  END PrintAt ;

(*******************************************************************************
*                                                                             *
* RechenRoutinen ...                                                          *
*                                                                             *
*******************************************************************************)

  PROCEDURE h ( x, y, z : INTEGER ) : INTEGER ;
    VAR
      a, b, c : REAL ;
  BEGIN
    a := ZentrumPunkt.x - FLOAT ( x ) ;
    b := ZentrumPunkt.y - FLOAT ( y ) ;
    c := ZentrumPunkt.z - FLOAT ( z ) ;
    RETURN TRUNC ( sqrt ( a * a + b * b + c * c ) ) ;
  END h ;

  PROCEDURE o ( a,b : Vector ) : REAL ;
  BEGIN
    RETURN a.x * b.x + a.y * b.y + a.z * b.z ;
  END o ;

  PROCEDURE x ( VAR r : Vector ; a,b : Vector ) ;
  BEGIN
    r.x := a.y * b.z - a.z * b.y ;
    r.y := a.z * b.x - a.x * b.z ;
    r.z := a.x * b.y - a.y * b.x ;
  END x ;

  PROCEDURE p ( VAR r : Vector ; a,b : Vector ) ;
  BEGIN
    r.x := a.x + b.x ;
    r.y := a.y + b.y ;
    r.z := a.z + b.z ;
  END p ;

  PROCEDURE m ( VAR r : Vector ; a,b : Vector ) ;
  BEGIN
    r.x := a.x - b.x ;
    r.y := a.y - b.y ;
    r.z := a.z - b.z ;
  END m ;

  PROCEDURE t ( VAR r : Vector ; o : Vector ; a : REAL ) ;
  BEGIN
    r.x := o.x * a ;
    r.y := o.y * a ;
    r.z := o.z * a ;
  END t ;

(*******************************************************************************
*                                                                             *
* Projection initialisieren                                                   *
*                                                                             *
*******************************************************************************)

  PROCEDURE IntQ ;
    VAR
      v, w : Vector ; a : REAL ; i : INTEGER ;
  BEGIN
    i := MyStringInfo [ 0 ].longInt ;
    IF ( i > 100 ) OR ( i < 3 ) THEN i := 8 ; END ;
    step := 360 / i ;
    WITH Normale1 DO x := -1. ; y := 0. ; z :=  0. ; END ;
    WITH Normale2 DO x :=  0. ; y := 0. ; z := -1. ; END ;
    a := FLOAT ( 90 - MyStringInfo [ 1 ].longInt ) / 57.296 ;
    v.x := 0. ;
    v.y := sin ( a ) ;
    v.z := cos ( a ) ;
    x ( Normale2, v, Normale1 ) ;
    t ( ZentrumPunkt, v, -2000. ) ;
    a := FLOAT ( 90 - MyStringInfo [ 2 ].longInt  ) / 57.296 ;
    t ( w, Normale2, sin ( a ) ) ;
    t ( v, Normale1, cos ( a ) ) ;
    p ( v, v, w ) ;
    t ( w, Normale2, - cos ( a ) ) ;
    t ( Normale1, Normale1, sin ( a ) ) ;
    p ( Normale1, Normale1, w ) ;
    Normale2 := v ;
    x ( BildEbene.n, Normale2, Normale1 ) ;
    t ( BildPunkt, BildEbene.n, 1500. ) ;
    m ( BildPunkt, ZentrumPunkt, BildPunkt ) ;
    BildEbene.k := - o ( BildPunkt, BildEbene.n ) ;
    WITH AbstandPunkt DO
      x := Normale1.x * 319. + Normale2.x * 255. ;
      y := Normale1.y * 319. + Normale2.y * 255. ;
      z := Normale1.z * 319. + Normale2.z * 255. ;
    END ;
    m ( AbstandPunkt, BildPunkt, AbstandPunkt ) ;
    AbstandEbene1.n := Normale1 ;
    AbstandEbene1.k := - o ( AbstandEbene1.n, AbstandPunkt ) ;
    AbstandEbene2.n := Normale2 ;
    AbstandEbene2.k := - o ( AbstandEbene2.n, AbstandPunkt ) ;
    LichtPunkt := AbstandPunkt ;
  END IntQ ;

(*******************************************************************************
*                                                                             *
* Projektion                                                                  *
*                                                                             *
*******************************************************************************)

  PROCEDURE Proj ( VAR x, y, z : INTEGER ) ;
    VAR a, g : Vector ; s : REAL ;
  BEGIN
    a.x := FLOAT ( x ) ;
    a.y := FLOAT ( y ) ;
    a.z := FLOAT ( z ) ;
    m ( g, ZentrumPunkt, a ) ;
    s := - ( o ( BildEbene.n, ZentrumPunkt ) + BildEbene.k ) / o ( BildEbene.n, g ) ;
    t ( a, g, s ) ;
    p ( a, a, ZentrumPunkt ) ;
    x := TRUNC (o ( AbstandEbene1.n, a ) + AbstandEbene1.k ) ;
```

```
    y := TRUNC (o ( AbstandEbene2.n, a ) + AbstandEbene2.k ) ;
  END Proj ;

  PROCEDURE ConQ ( x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4 : INTEGER ) ;
    VAR
      dv1, dv2 : Vector ; light : INTEGER ; dreal : REAL ;
  BEGIN
    IF gestalt = schatten THEN
      dv1.x := FLOAT ( x2 - x1 ) ;
      dv1.y := FLOAT ( y2 - y1 ) ;
      dv1.z := FLOAT ( z2 - z1 ) ;
      dv2.x := FLOAT ( x3 - x2 ) ;
      dv2.y := FLOAT ( y3 - y2 ) ;
      dv2.z := FLOAT ( z3 - z2 ) ;
      IF sqrt ( dv2.x * dv2.x + dv2.y * dv2.y + dv2.z * dv2.z ) = 0. THEN
        dv2.x := FLOAT ( x1 - x4 ) ;
        dv2.y := FLOAT ( y1 - y4 ) ;
        dv2.z := FLOAT ( z1 - z4 ) ;
      END ;
      x ( dv1, dv1, dv2 ) ;
      dv2.x := FLOAT ( x2 ) ;
      dv2.y := FLOAT ( y2 ) ;
      dv2.z := FLOAT ( z2 ) ;
      m ( dv2, dv2, LichtPunkt ) ;
      dreal := o ( dv1, dv2 ) ;
      dreal := dreal / sqrt ( dv1.x * dv1.x + dv1.y * dv1.y + dv1.z * dv1.z ) ;
      dreal := dreal / sqrt ( dv2.x * dv2.x + dv2.y * dv2.y + dv2.z * dv2.z ) ;
      IF ( dreal > 0.99 ) AND ( dreal < 1.01 ) THEN
        dreal := 0. ;
      ELSE
        dreal := 1.57 - arctan ( dreal / ( sqrt ( 1. - dreal * dreal ) ) ) ;
      END ;
      light := TRUNC ( dreal / 3.14 * 12. ) + 4 ;
      SetAPen ( MyRastPortPtr, light ) ;
    END ;
    Proj ( x1, y1, z1 ) ;
    Proj ( x2, y2, z2 ) ;
    Proj ( x3, y3, z3 ) ;
    Proj ( x4, y4, z4 ) ;
    Quad ( x1, y1, x2, y2, x3, y3, x4, y4 ) ;
  END ConQ ;

(*****************************************************************************
*                                                                            *
* Rotationskörper aufbauen                                                   *
*                                                                            *
*****************************************************************************)

  PROCEDURE BuildIt ( Last : INTEGER ) ;
    VAR j, i, l : INTEGER ;
  BEGIN
    SetAPen ( MyRastPortPtr, 0 ) ;
    RectFill ( MyRastPortPtr, 0, 0, 639, 511 ) ;
    SetAPen ( MyRastPortPtr, 1 ) ;
    IntQ ;
    FOR j := 0 TO Last + 1 DO
      RO [ j ] [ 0 ] := 0 ;
      RO [ j ] [ 1 ] := O [ j ] [ 0 ] ;
    END ;
    i := step ;
    WHILE i <= 180 DO
      RS := RO ;
      max := TRUNC ( sin ( FLOAT ( i ) / 57.296 ) * 16384. ) ;
      may := TRUNC ( cos ( FLOAT ( i ) / 57.296 ) * 16384. ) ;
      FOR j := 0 TO Last + 1 DO
        RO [ j ] [ 0 ] := ( O [ j ] [ 0 ] * max ) / 16384 ;
        RO [ j ] [ 1 ] := ( O [ j ] [ 0 ] * may ) / 16384 ;
      END ;
      FOR j := 0 TO Last DO
        l := j + 1 ;
        C [ j ] [ 12 ] := h ( RO [ j ] [ 0 ], RO [ j ] [ 1 ], O [ j ] [ 1 ] ) ;
        C [ j ] [ 13 ] := h ( RO [ l ] [ 0 ], RO [ l ] [ 1 ], O [ l ] [ 1 ] ) ;
        IF C [ j ] [ 12 ] > C [ j ] [ 13 ] THEN
          dint := C [ j ] [ 12 ] ;
          C [ j ] [ 12 ] := C [ j ] [ 13 ] ;
          C [ j ] [ 13 ] := dint ;
        END ;
        C [ j ] [ 00 ] := RO [ j ] [ 0 ] ;
        C [ j ] [ 01 ] := RO [ j ] [ 1 ] ;
        C [ j ] [ 02 ] :=  O [ j ] [ 1 ] ;
        C [ j ] [ 03 ] := RO [ l ] [ 0 ] ;
        C [ j ] [ 04 ] := RO [ l ] [ 1 ] ;
        C [ j ] [ 05 ] :=  O [ l ] [ 1 ] ;
        C [ j ] [ 06 ] := RS [ l ] [ 0 ] ;
        C [ j ] [ 07 ] := RS [ l ] [ 1 ] ;
        C [ j ] [ 08 ] :=  O [ l ] [ 1 ] ;
        C [ j ] [ 09 ] := RS [ j ] [ 0 ] ;
        C [ j ] [ 10 ] := RS [ j ] [ 1 ] ;
        C [ j ] [ 11 ] :=  O [ j ] [ 1 ] ;
      END ;
      IF gestalt # draht THEN
        FOR j := 0 TO Last - 1 DO
          FOR l := j + 1 TO Last DO
            IF ( C [ j ] [ 12 ] < C [ l ] [ 12 ] ) OR
               ( ( C [ j ] [ 12 ] = C [ l ] [ 12 ] ) AND
               ( C [ j ] [ 13 ] < C [ l ] [ 13 ] ) )
            THEN
              C [ 63 ] := C [ l ] ;
              C [ l ] := C [ j ] ;
              C [ j ] := C [ 63 ] ;
            END ;
          END ;
        END ;
      END ;
      FOR j := 0 TO Last DO
        ConQ (
          C [ j ] [ 00 ], C [ j ] [ 01 ], C [ j ] [ 02 ],
          C [ j ] [ 03 ], C [ j ] [ 04 ], C [ j ] [ 05 ],
          C [ j ] [ 06 ], C [ j ] [ 07 ], C [ j ] [ 08 ],
          C [ j ] [ 09 ], C [ j ] [ 10 ], C [ j ] [ 11 ] ) ;
        ConQ (
          - C [ j ] [ 00 ], C [ j ] [ 01 ], C [ j ] [ 02 ],
          - C [ j ] [ 03 ], C [ j ] [ 04 ], C [ j ] [ 05 ],
          - C [ j ] [ 06 ], C [ j ] [ 07 ], C [ j ] [ 08 ],
          - C [ j ] [ 09 ], C [ j ] [ 10 ], C [ j ] [ 11 ] ) ;
      END ;
      INC ( i, step ) ;
    END ;
    WHILE 6 IN LeftButton DO ; END ;
  END BuildIt ;

(*****************************************************************************
*                                                                            *
* Menü ...                                                                   *
*                                                                            *
*****************************************************************************)

  PROCEDURE DoMenu ;
  BEGIN
    Move ( MyRastPortPtr, 0, 0 ) ;
    ClearScreen ( MyRastPortPtr ) ;
    SetAPen ( MyRastPortPtr, 3 ) ;
    PrintAt ( 0, 30, "Schritte" ) ;
    PrintAt ( 0, 86, "Kippwinkel" ) ;
    PrintAt ( 0, 142, "Drehwinkel" ) ;
    PrintAt ( 0, 198, "Darstellung" ) ;
    PrintAt ( 0, 366, "Aktion" ) ;
    MakeGadget (  20, 0, "", 2 ) ;
    MakeGadget (  76, 1, "", 2 ) ;
    MakeGadget ( 132, 2, "", 2 ) ;
    MakeGadget ( 188, 3, "  Draht   ", 1 ) ;
    MakeGadget ( 244, 4, " Massiv   ", 1 ) ;
    MakeGadget ( 300, 5, "Schatten  ", 1 ) ;
    Turn ( gestalt ) ;
    MakeGadget ( 356, 6, "   NEU    ", 0 ) ;
    MakeGadget ( 412, 7, "    OK    ", 0 ) ;
    MakeGadget ( 468, 8, "   ENDE   ", 0 ) ;
    MakeGadget ( 0, 9, "", 3 ) ;
    SetAPen ( MyRastPortPtr, 2 ) ;
    RectFill ( MyRastPortPtr, 320, 0, 639, 511 ) ;
    SetAPen ( MyRastPortPtr, 3 ) ;
    Move ( MyRastPortPtr, 320 + O [ 0 ] [ 0 ], 256 + O [ 0 ] [ 1 ] ) ;
    FOR i := 1 TO j - 1 DO
      Draw ( MyRastPortPtr, 320 + O [ i ] [ 0 ], 256 + O [ i ] [ 1 ] ) ;
    END ;
  END DoMenu ;

(*****************************************************************************
*                                                                            *
* Supervise ( MAIN )                                                         *
*                                                                            *
*****************************************************************************)

BEGIN
  IF OpenDisplay () THEN
    Mybuffer [ 0 ] := "010" ;
    Mybuffer [ 1 ] := "030" ;
    Mybuffer [ 2 ] := "000" ;
    gestalt := draht ;
    j := 0 ;
    DoMenu ;
    MyIntuiMessagePtr := NIL ;
```

Rotationskörper

```
      LOOP
        ax := MyScreenPtr^.mouseX ;
        ay := MyScreenPtr^.mouseY ;
        IF ( ax >= 320 ) AND ( j > 0 ) AND ( j < 63 ) THEN
          SetDrMd ( MyRastPortPtr, DrawModeSet { complement } ) ;
          Move ( MyRastPortPtr, ex, ey ) ;
          Draw ( MyRastPortPtr, ax, ay ) ;
          SetDrMd ( MyRastPortPtr, jam1 ) ;
        END ;
        WaitPort ( MyWindowPtr^.userPort ) ;
        IF ( ax >= 320 ) AND ( j > 0 ) AND ( j < 63 ) THEN
          SetDrMd ( MyRastPortPtr, DrawModeSet { complement } ) ;
          Move ( MyRastPortPtr, ex, ey ) ;
          Draw ( MyRastPortPtr, ax, ay ) ;
          SetDrMd ( MyRastPortPtr, jam1 ) ;
        END ;
        MyIntuiMessagePtr := GetMsg ( MyWindowPtr^.userPort ) ;
        MyIntuiMessage := MyIntuiMessagePtr^ ;
        ReplyMsg ( MyIntuiMessagePtr ) ;
        MyGadgetPtr := MyIntuiMessage.iAddress ;
        MyClass := MyIntuiMessage.class ;
        IF gadgetDown IN MyClass THEN
          CASE MyGadgetPtr^.gadgetID OF
          3 :
            Select ( draht )
          |
          4 :
            Select ( filled )
          |
          5 :
            Select ( schatten )
          |
          6 :
            j := 0 ;
            RectFill ( MyRastPortPtr, 320, 0, 639, 511 ) ;
            SetAPen ( MyRastPortPtr, 2 ) ;
            RectFill ( MyRastPortPtr, 320, 0, 639, 511 ) ;
            SetAPen ( MyRastPortPtr, 3 )
          |
          7 :
            IF j > 1 THEN
              FOR i := 0 TO 9 DO
                dint := RemoveGadget ( MyWindowPtr, ADR ( MyGadget [ i ] ) ) ;
              END ;
              BuildIt ( j - 2 ) ;
              DoMenu ;
            ELSE
              DisplayBeep ( MyScreenPtr ) ;
            END
          |
          8 :
            EXIT
          |
          9 :
            IF j < 63 THEN
              O [ j ] [ 0 ] := MyScreenPtr^.mouseX - 320 ;
              O [ j ] [ 1 ] := MyScreenPtr^.mouseY - 256 ;
              ex := 320 + O [ j ] [ 0 ] ;
              ey := 256 + O [ j ] [ 1 ] ;
              IF j # 0 THEN
                Move ( MyRastPortPtr,
                  320 + O [ j - 1 ] [ 0 ], 256 + O [ j - 1 ] [ 1 ] ) ;
                Draw ( MyRastPortPtr, ex, ey ) ;
              ELSE
                dlong := WritePixel ( MyRastPortPtr, ex, ey ) ;
                Move ( MyRastPortPtr, i, 1 ) ;
              END ;
              INC ( j ) ;
            ELSE
              DisplayBeep ( MyScreenPtr ) ;
            END
          |
        ELSE
        END ;
      END ;
    END ;
    CloseDisplay ;
  END ;
END Rod.

(**************************************************************************************)
```

Rotationskörper

JINTECK-VERSAND, Auwiesenweg 3, D-8049 Unterbruck/Fahrenzhausen, Tel. 0 81 33/801-3



JINTECH

SPAß am Home-Computer SPAß am Home-Computer

Modell	Kaufpreis/DM				Versand/DM
Diskettenlaufwerke					
3 A-Senator	225,--				10,-
Master 5 A-1	286,--				
Master 3S	225,--				
Master 5S	295,--				8,--
Beschleuniger					
Oceanic Turbo Dos					8,--
Mouse					
Oceanic Mouse Amiga	65,--				
Oceanic Mouse Atari	65,--				
Oceanic Mouse C64	54,--				
Komplettes Spar-Paket	**P1**	**P2**	**P3**	**P4**	10,--
KY-1000H	x	x	x	x	
Turbo Dos	x	x	x	x	
Joystick		x		x	
Mouse			x	x	
DM	204,--	224,--	253,--	283,--	

* Aufpreis für ein Orginal C64 zu jedem Paket: DM 282,--
+ Versand DM 7,--

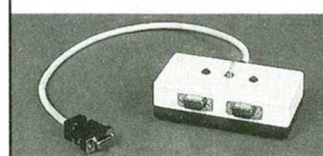
Modellbeschreibung:

KY-1000 H:	5,25"-Diskettenlaufwerke für C64 Commodore-Computer. Speicherkapazität 174 K (GCR-formatiert), 36 Spuren, 48 TPI-Dichte, Erweitungsstecker für DOS
Senator:	3,5"-Diskettenlaufwerke für Amiga-Computer, Speicherkapazität 1 MByte (MFM-formatiert), 160 Spuren, 135 TPI-Dichte
Master 3A II:	3,5"-Diskettenlaufwerke intern für Amiga 2000-Computer (FDD)
Master 5A-1:	5,25"-Diskettenlaufwerke für Amiga-Computer, Speicherkapazität 1 MByte (MFM-formatiert), 160 Spuren, 96 TPI-Dichte, 40/80-Schalter
Master 3S:	3,5"-Diskettenlaufwerke für Atari ST-Computer, Speicherkapazität 1 MByte (MFM-formatiert), 160 Spuren, 96 TPI-Dichte, 40/80-Schalter
Master 5S:	5,25"-Diskettenlaufwerke für Atarie ST-Computer, Speicherkapazität 1 MByte, (MFM-formatiert), 160 Spuren, 96 TPI-Dichte, 40/80-Schalter
Turbo DOS-Cartridge:	voll kompatibel zu OC-118N und KY-1000 H 3 x schneller als 1541, kein Laden nötig
JT-M01-Maus:	voll kompatibel zu 64/128
JT-M02-Maus:	Ersatz für Atari ST-Maus
JT-M03-Maus:	Ersatz für Amiga-Maus

* **Ein Anruf bei uns genügt !**
* **Das Paket wird Ihnen innerhalb 3 Tagen zugestellt !**
* **Sie zahlen erst bei Paketzustellung !**
* **1/2 Jahr Garantie !**
* **Reparatur dauert nur 3 Tage !**



Klick!! Und der Horror mit der Umstöpselei zwischen Maus und Joystick, Maus und Kopierschutz etc. hat ein Ende!



Vorgestellt in Amiga 9/89.
Der Adapter ist kompatibel zum Industriestandard - anschließbar an: AMIGA 500/1000/2000, Commodore 64, 128, C16, Plus4, MSX, u.a.
Solide Industriequalität mit zwei LED's.
Preis nur DM **45,-**

SPECIAL OFFER:

Laufwerk Amiga 3,5" kompl.	DM 239,-
Laufwerk Amiga 5,25" kompl.	DM 269,-
Speichererweiterung A 500 512 KB (abschaltbar + Uhr)	DM 198,-

NEU! NEU! NEU! NEU! NEU! NEU!
Schützen Sie Ihren "AMIGA" vor fremden Zugriffen!!!
Durch die Hardware von H+W ist das Problem gelöst.
Rufen Sie uns an, wir helfen weiter! **DM 29,-**

H + W Computer + Zubehör
Egonstr. 13, 4650 Gelsenkirchen 2
Bestellservice: 02 09-6 74 62

Amiga, Commodore sind eigetragene Warenzeichen

BEST – MODEMS

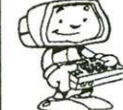
Best 2400EC MNP-5 649,-
Automatische MNP-Fehlerkorrektur.
Durch automatische Datenkompression erreicht es ca. 4800 Baud.
300, 1200, 2400 Baud
V.21, V.22, V.22bis, Bell 103 & 212 A

Best 2400 Plus 447,-
Echtes 4-Speed-vollduplex Modem
300, 1200, 75/1200, 1200/75, 2400 Baud
V.21, V.22, V.22bis, V.23,
Bell 103 & 212 A

Best 1-2-3 329,-
300, 1200, 75/1200, 1200/75 Baud
V.21, V.22, V.23, Bell 103 & 212 A

Best 1200 Plus 289,-
300, 1200 Baud
V.21, V.22, Bell 103 & 212 A

RS-232 Kabel	**20,-**
Konverterstecker auf TAE-6	**12,-**





Hayes kompatibel

Bei V.23 BTX zum Ortstarif

Die Inbetriebnahme dieser Modems am öffentlichen Postnetz der BRD einschl. West-Berlin ist zur Zeit verboten und unter Strafe gestellt.

Alfonso Sanz Software & Beratung
Rabanus Maurus Str. 8 / 6415 Petersberg
Telefon: (0661)-61748

AmigaBASIC – Programmieren mit Erfolg

Mit WENN, aber ohne GOTO

Kursfahrplan
Teil 1: Basic zum Kennenlernen
Teil 2: Dem Spaghetti-Code keine Chance
Teil 3: Vom Strich zum Bild
Teil 4: Dateien über Dateien
Teil 5: Unter dem Programm: ein Unterprogramm

Im ersten Teil des Kurses haben wir Ihnen AmigaBASIC (hoffentlich) schmackhaft gemacht, zumindest dürfte Ihnen unsere Aufgabe, die wir gestellt hatten und deren Auflösung wir am Ende des Kurses bringen werden, den Einstieg in diese neue Programmierumgebung erleichtert haben. Nun aber geht es weiter, wir beschäftigen uns heute mit einer der besten Eigenschaften des AmigaBASIC – der strukturierten Programmierung.

AmigaBASIC bietet nicht nur eine Unmenge an Befehlen, es bietet darüber hinaus die Möglichkeit, Programme so zu organisieren, daß sie überschaubar werden. In anderen, systemnäheren Hochsprachen wie Pascal, Modula-2 oder C findet man diese Strukturierung, also die Möglichkeit, Programme überschaubar zu machen, allerorts, in BASIC-Dialekten dagegen nicht allzu häufig. AmigaBASIC sowie auch GFA-BASIC Amiga bilden da wohltuende Ausnahmen. Der Begriff 'strukturierbar' ist eigentlich dehn- und ziehbar, wir wollen im AmigaBASIC jedoch einfach dabei bleiben, Programme, die in einzelne, durchschaubare Teilprogramme aufgegliedert sind, so zu benennen.

Mal hierhin, mal dorthin – springen will gelernt sein

Machen wir uns diese Haupteigenschaft des AmigaBASIC doch einmal deutlich:
Wenn Sie Aufsteiger sind und zuvor einen 8-Bit-Computer wie C-64 oder CPC Ihr eigen genannt haben, so haben Sie sicherlich auch die dort vorkommende BASIC-Sprache in Augenschein genommen (diejenigen, die direkt mit dem Amiga eingestiegen sind, werden jetzt zwar nicht alles kennen, sollten jedoch zum besseren Verständnis diese Passage trotzdem weiterlesen). Bevor dort eine Zeile als Programmteil angenommen wurde, mußte Sie durch eine Zeilennummer gekennzeichnet werden. Diese Zeilennummern mußten in aufsteigender Form vorhanden sein, damit der jeweilige Interpreter, also der Betriebssystemteil, der mit der Umwandlung der BASIC-Worte in für den Prozessor verständliche Codes beschäftigt ist, die einzelnen Zeilen als Programm erkannte.
Der jeweilige BASIC-Programmierer bekam diesen Umstand eigentlich kaum mit, er war seine Programmierumgebung gewöhnt, hier war sein 'Zuhause'. Da er nichts bemerkte, sahen seine Erstlingsprogramme dementsprechend aus. Sie glauben es nicht?
Sehen wir uns ein kleines Beispiel an:

100 w$ = INKEY$
110 IF w$ = "1" THEN GOTO 200
120 IF w$ = "2" THEN GOTO 300
130 IF w$ = "3" THEN GOTO 400
140 GOTO 100

Eine einfache Tastaturabfrage – und doch 'starker Tobak', wie der Nordseefischer sagt. Es gibt hier nämlich schon ein paar potentielle Fehlerquellen:
Wenn dieser Teil am Anfang eines Programms gesetzt wird, kann man nie sicher sein, ob der Platz zwischen Zeile 200 und 300 ausreichend für den entsprechenden Programmteil ist. Sie kennen dies sicher vom Kochen: Ein bißchen hiervon, ein bißchen davon, etwas dieses, manches jenes – und schon hat man sich ein nettes Süppchen eingebrockt.
Die eventuelle zweite Fehlerquelle: Wer weiß denn, ob die Tasten 1 bis 3 nicht irgendwann für etwas anderes nötig werden? Dann geht es erst richtig los: Das Programm springt mit seinen GOTOs überall hin, nur nicht in die richtige Routine.
Ich will hier nicht den Teufel an die Wand malen, es gibt ausgefuchste BASIC-Programmierer, die selbst 1000 Zeilen mühelos überschauen (meine Hochachtung!!!), jedoch Anfänger verlieren nach solchen 'Sprungorgien' schnell den Überblick und, was noch schlimmer ist, die Lust am Programmieren.

AmigaBASIC braucht keine Nummern für die Zeilen, Namen reichen

Gehen wir etwas näher auf das kleine Programm ein: Die Tasten 1 bis 3 sollen eine bestimmte Aktion auslösen, dazu müssen Sie irgendwie abgefragt werden. In AmigaBASIC und anderen Dialekten gibt es dazu die Funktion 'INKEY$'. Sie macht eigentlich nichts anderes, als das zuletzt eingegebene Zeichen einzulesen und für Überprüfungen vorzubereiten.
Die Zeile *w$* = INKEY$ weist der Stringvariablen w$ das zuletzt eingegebene Zeichen zu. (Wer jetzt Schwierigkeiten beim Begriff Stringvariable hat, sollte am Ende dieses Kursteils ins Glossar und in sein BASIC-Handbuch schauen!).
Ein kurzes Beispiel dazu:

100 w$ = INKEY$
110 IF w$< >"e" GOTO 100

Toll... Genau das, was wir nicht wollen: erstens zwei Zeilennummern, zweitens ein GOTO. Lassen wir die Zeilennummern weg, wie wir es schon mehrmals gehört haben, erhalten wir:

w$ = INKEY$
GOTO 100

GOTO 100? Was für eine 100? Starten wir das Programm, bricht 'good old Amiga' mit Blitzlicht und Getöse ab und meint, er hätte ein 'Undefined label' gefunden. Was also tun?
AmigaBASIC bietet einen fast genialen Ersatz für diese Probleme, eigentlich bringt es uns diesen Ersatz gleich vor Augen: **LABEL.** Was ist ein Label? Es ist eine Markierung oder Marke. Eine Markierung, die vom Programm akzeptiert und benutzt werden kann. Wie sieht nun so ein Label aus?
Grundsätzlich akzeptiert das BASIC des Amiga jede Bezeichnung als Label, zum Beispiel das Wort 'Programm'. Eines müssen wir jedoch beachten: Label müssen am Ende mit einem Doppelpunkt gekennzeichnet werden, um als solche erkannt zu werden. Also schreiben wir nach Anweisung nicht 'Programm' sondern 'Programm:'.

```
SCREEN 2,320,256,2,1:WINDOW 2,"Test",,2,2
WHILE INKEY$<>UCASE$("e")
    CLS:Zaehler = 29:vg=0:hg=1
    WHILE Zaehler > 1
        LOCATE Zaehler,1:COLOR vg,hg
        vg=INT(RND*4):hg=INT(RND*4)
        IF vg=hg THEN vg=vg-1:IF vg=-1 THEN vg=1
        PRINT "Wir befinden uns in Zeile ";Zaehler
        Zaehler = Zaehler - 1
        FOR n=1 TO 1000:NEXT n
    WEND
WEND
WINDOW CLOSE 2: SCREEN CLOSE 2
```

Abb. 1: Der Einsatz von WHILE/WEND als Zählerschleife ist in diesem Programm zu sehen

Wo setzt man nun Label am geschicktesten ein? Am besten immer vor einem bestimmten Programmteil. Werden Label nicht angesprungen, haben sie immerhin den Zweck der Kennzeichnung und die lästig-langen REM-Zeilen gehören ein für alle Mal der Vergangenheit an. (Manche 'Programmierer' machen gerne einen 20zeiligen REM-Vorspann für ein zehnzeiliges 'Hallo Welt'-Programm – etwas, was uns hier nicht einfallen sollte!)

Unser kleines Beispielprogramm sähe nach Einsatz eines Labels dann so aus:

Abfrage:
w$ = INKEY$
IF w$< > "e" GOTO Abfrage

Aha!! So geht das! Wir wissen also jetzt, was ein Label bewirken kann:

1.) *Sprungmarke, um an bestimmte Programmteile zu gelangen, und*
2.) *Kennzeichnung dieses Programmteils.*

Für Labels kann man grundsätzlich alle Namen nehmen, die einem in den Sinn kommen, allerdings müssen Labelnamen **immer** mit einem Buchstaben beginnen und mit Doppelpunkt enden. Man kann alle Zeichen zwischen A(a) und Z(z) benutzen (außer den deutschen Umlauten äüöß) und alle Ziffern zwischen 1 und 9. Namen wie 'Unterprogram_1' werden allerdings vom System reklamiert, der Unterstrich gehört nicht zum Repertoire des Labelinterpreters.

Eigentlich könnten wir ja nun schon zufrieden sein; wir wissen, wie man ohne Zeilennummern programmiert und mit Hilfe von Markierungen (im weiteren werden wir sie immer Label nennen) auf Programmabschnitte zugreift. Leider besitzt die Sache oder eben unser 'Prográmmchen' noch einen großen Schönheitsfehler: das GOTO.

Je höher man springt, desto tiefer fällt man (abgewandeltes Sprichwort)

Der GOTO-Befehl kommt in jedem BASIC-Dialekt vor, sei es auf den 'Achtbittern', sei es auf PCs, Amigas, Ataris oder WEWENOWAs (WErWEißNOchWAs). Auch in der 'höheren Programmiersprache' Pascal existiert er. Komischerweise bekommt fast jeder, der sich in der strukturierten Programmierung auskennt, beim alleinigen Nennen des Befehls rote Punkte im Gesicht. Nein, eigentlich nicht komischerweise, eigentlich mit einiger Berechtigung.

```
REM Programm BASICKURS_002.002
REM ---------------------------
'
'
DIM Zeichen2$(30)
SCREEN 2,640,256,2,2:WINDOW 2,"Von klein nach gro3!",,2,2

WHILE flag=0
   CLS:Eingabe$="":Ausgabe$="":Zeichen1$="":Zeichen2$="":merker=0
   PRINT"Bitte geben Sie Ihren Namen in Kleinschrift ein, trennen"
   PRINT"Sie bitte Vor- und Zunamen durch ein Leerzeichen. Wenn  "
   PRINT"Sie das Programm beenden wollen geben Sie nur 'x' ein   "
   PRINT"und drucken die ENTER-Taste.
   PRINT
   INPUT"Ihr Name ";Eingabe$
   IF LEN(Eingabe$) > 1 THEN
      Zeichen1$=UCASE$(MID$(Eingabe$,1,1))
      MID$(Eingabe$,1,1)=Zeichen1$
      FOR n=1 TO LEN(Eingabe$)
          Zeichen2$(n)=MID$(Eingabe$,n,1)
          IF Zeichen2$(n)=CHR$(32) THEN merker=n+1
      NEXT n
      MID$(Eingabe$,merker,1)=UCASE$(MID$(Eingabe$,merker,1))
      PRINT Eingabe$
      PRINT"Bitte Taste drucken!!":WHILE INKEY$="":WEND
   ELSE
      flag=1
   END IF
WEND

WINDOW CLOSE 2:SCREEN CLOSE 2
CLS:LIST:END
```

Abb.2: Ein Beispiel für die Verwendung von UCASE$

GOTO verführt jeden irgendwann dazu, Spaghetti-Code zu programmieren. Warum? Nun, schauen wir uns unser Beispiel vom Anfang dieses Textes noch einmal an:

100 w$ = INKEY$
110 IF w$ = "1" THEN GOTO 200
120 IF w$ = "2" THEN GOTO 300
130 IF w$ = "3" THEN GOTO 400
140 GOTO 100

Wir hatten anfangs Bedenken, daß der Platz zwischen den beiden großen Zeilennummern nicht reicht oder, noch schlimmer, man hinterher bemerkt, daß eigentlich alles ganz anders laufen müßte...

GOTO, wie gesagt, verführt, und zwar zu unkontrollierbaren Sprüngen. Keine Angst, der Amiga springt nicht nach drei GOTOs aus dem Fenster, aber Sie als Programmerzeuger wissen irgendwann vor lauter GOTOs nicht mehr, wohin Sie gehen sollen, das einzige, was dann noch ein GOTO macht, ist Ihr Programm, und zwar ins Nirwana.

Vielleicht ist dies alles noch nicht so einleuchtend für Sie, und Sie fragen sich, warum gibt es GOTO eigentlich, wenn es doch keiner mag? Nun, eigentlich ist der GOTO-Befehl ein Relikt aus den Anfängen des BASIC, als die meisten Programme noch aus UPOs (unidentifizierbare prozessorstörende Objekte) bestanden und das Wort Struktur höchstens bei Gipsformen Anwendung fand. Inzwischen sind die BASIC-Dialekte so gut wie ausgereift, AmigaBASIC erfüllt fast alle Wünsche, und es existieren Befehle und Funktionen, von denen man vor gar nicht so langer Zeit noch nicht einmal zu träumen wagte.

Gehen wir ans Eingemachte: Wir wollen kein GOTO, aber was können wir nehmen?

AmigaBASIC stellt uns zum Beispiel die Befehle WHILE und WEND zur Verfügung. WHILE und WEND gehören zusammen wie eineiige Zwillinge. Übersetzen kann man die Befehle mit 'solange' und 'wiederhole'.

Zum besseren Verständnis ein Beispiel:

WHILE
INKEY$< > "e":WEND

Laden Sie doch einmal AmigaBASIC (ich hoffe, Sie besitzen schon eine startfähige BASIC-Diskette mit allen benötigten Dateien), und gehen Sie mit dem Cursor ins OUTPUT-Window. Geben Sie die obenstehende Zeile im Direktmodus ein, und drücken Sie die ENTER-Taste [< -d9-]. Der Cursor verschwindet so lange, bis Sie die Taste E gedrückt haben. Übrigens: SHIFT und E funktioniert nicht, dazu später mehr.

Ach ja, Sie haben soeben festgestellt, daß unser 'Prográmmchen' mit Label und GOTO nicht mehr benötigt wird.

WHILE und WEND bilden eine Schleife, ähnlich wie die bekanntere FOR-NEXT-Kombination. Übersetzt würde die Zeile so lauten:

Solange die Taste 'e' nicht gedrückt wurde, wiederhole diese Zeile.

WHILE und WEND können nicht nur solche Kunststückchen, mit ihnen kann man auch Zähler ablaufen lassen:

Zaehler = 20 WHILE Zaehler > 1
LOCATE Zaehler,1
PRINT"Dies ist Zeile ";
Zaehler
Zaehler = Zaehler-1
WEND

Das Programm schreibt einen Text von unten nach oben auf den Bildschirm. In Abbildung 1 sehen Sie eine etwas verbesserte Version, die auch schon ein bißchen mit der Farbe spielt, aber wirklich nur ein bißchen.

Die dort vorkommenden SCREEN- und WINDOW-Befehle können weggelassen werden, wenn Sie mit der A500-Grundkonfiguration von 512 kByte arbeiten, da diese Befehle sehr viel Speicherplatz benötigen.

Mit WHILE und WEND lassen sich wirklich schon einige Programmprobleme lösen, unsere berühmte Tastaturabfrage läßt sich so ohne weiteres mit einem Programm verknüpfen, ohne auch nur einen Sprung zu machen:

WHILE INKEY$
< > UCASE$("e")
Programm...
Programm...
Programm...
WEND

Huch! Schon wieder etwas Neues! UCASE$ – noch nie gehört!

Keine Angst, die Erklärung erfolgt sofort:

Soeben hatten wir bemerkt, daß unser 'Prográmmchen' zwar auf die Eingabe des kleinen 'e' reagiert, sich bei geshiftetem 'E' jedoch taub stellt. Um zu gewährleisten, daß Groß- und Kleinschrift bemerkt wird, gibt es die Funktion UCASE$. Sie wandelt die vorhandenen Stringvariable in Großschrift um

w$ = "abc":w$ = UCASE$(w$)

gibt nachher die Zeichenfolge 'ABC' aus.

*UCASE$* läßt sich in sehr vielen Programmen sinnvoll einsetzen, vor allem dann wenn man auf Groß- und Kleinschrift achten will. Schauen Sie sich dazu auch mal Listing 2 an, ein kleines Beispiel zu UCASE$, in dem auch alle weiteren an dieser

Stelle genannten BASIC-Befehle Verwendung finden. Natürlich läßt sich so etwas noch eleganter programmieren, aber zum Ansehen reicht es doch wohl.

Doch zurück zu WHILE und WEND: Unser oben gezeigtes Listing wiederholt einen Programmteil so lange, bis die Taste [e] gedrückt wurde.

Wenn wir mit unserem neuen Wissen jetzt noch einmal unser allererstes Programm (das mit den vielen GOTOs) überdenken, wird uns schnell klar, daß selbst ein so kleines Modul eine gewisse Vorbereitungszeit braucht. Vom schnellen 'Drauflosprogrammieren' ist da bald keine Rede mehr. AmigaBASIC zwingt den Programmierer eigentlich durch seine vielen Sprachelemente, sich vorher Gedanken über die Ausführung seines Programms zu machen.

Wenn wir an diesem Punkt angelangt sind, bemerken wir schnell, daß wir für die Abfrage der Tasten etwas mehr brauchen. Ein Ansatz dieses 'etwas mehr' war schon da, der IF/THEN-Befehl. Doch er reicht uns in diesem Maße noch nicht.

```
Programm:
   flag1=0:Flag2=0:a=0
   WHILE flag1=0
      INPUT"Bitte drucken Sie Taste 1,2 oder 3 ";w$
      PRINT
      IF w$="1" THEN
         Flag2=1
      ELSEIF w$="2" THEN
         Flag2=2:
      ELSEIF w$="3" THEN
         Flag2=3
      ELSE
         PRINT"Sie haben nicht '1','2' oder '3' gewahlt!"
         PRINT
         PRINT"Nochmal (j/n)  ";:INPUT Antwort$
         IF UCASE$(Antwort$)="N" THEN flag1=1
      END IF
      IF flag1 <> 1 THEN GOSUB Variable:GOSUB Unterprogramm
   WEND
   CLS:END

Variable:
   IF Flag2=1 THEN
      a=1
   ELSEIF Flag2=2 THEN
      a=2
   ELSE
      a=3
   END IF
   RETURN

Unterprogramm:
   CLS:PRINT"Sie haben Taste ";a;" gedruckt!"
   RETURN
```

Abb.3: Mit IF…THEN und den verwandten Befehlen lassen sich Abfragen übersichtlich gestalten

Wenn ja, darf ich dann? Entscheidungen sind gar nicht schwer

IF und THEN dürften den meisten unter Ihnen geläufig sein, für diejenigen, die in BASIC noch keine große Erfahrung haben, hier ein kleiner Abriß:

IF und THEN werden grundsätzlich dann eingesetzt, wenn man anhand eines bestimmten Ereignisses einen bestimmten Ablauf des Programmes erreichen will. Genauer gesagt nennt man diese Umschreibung Entscheidung. Stellen Sie sich vor, Sie wollen in das Freibad (zur Zeit etwas illusorisch, aber was nicht ist…), dies hängt jedoch davon ab, wie das Wetter wird. Ist es sonnig und warm gehen Sie, ansonsten nicht. Im Programm sähe es so aus:

IF Wetter = sonnig THEN gehe schwimmen

IF Wetter = nicht sonnig THEN lese AMIGA DOS zu Hause.

Ich habe bewußt im zweiten IF/THEN eine Negation verwendet, sie fragt nämlich alle Eventualitäten ab. Hätten Sie statt 'nicht sonnig' 'regnerisch' eingesetzt, und es hätte geschneit, so hätten Sie trotzdem ins Schwimmbad gehen müssen, um Ihrer Bedingung Rechnung zu tragen.

Sie merken schon, IF und THEN nehmen es mit Ihrer Einstellung zur Programmierung ebenfalls hochgenau, auch hier würde eine 'Mal-eben-Programmierung' höchstwahrscheinlich in eine Nachtfehlersuche ausarten. Dabei bietet der IF/THEN-Verbund eine Menge an Variationsmöglichkeiten, denn zu ihm gesellen sich noch das ELSE (nicht die Else!!) und das END IF, und was es damit alles auf sich hat, sehen wir uns jetzt mal an.

WENN-DANN-SONST – oder wie oder wer oder was? Wir entscheiden eine ganze Menge

Um noch einmal auf unser obiges Beispiel zu kommen, hier haben wir zwei aufeinanderfolgende IF/THEN-Entscheidungen. Mit ein bißchen Geschick und unserem AmigaBASIC läßt sich dies auch noch vereinfachen.

Zu IF ('wenn') und THEN ('dann') gehören auch noch ELSE ('sonst'), ELSE IF ('sonst wenn') und END IF ('Ende wenn'). Diese deutschen Umschreibungen hören sich etwas komisch an, setzen Sie sie jedoch in komplette Umschreibungen einer Zeile ein, wird Ihnen schnell der Sinn derselben klarer. Wichtig ist dies vor allem dann, wenn man in der Planung eines Programms ist. Setzen wir unsere obige 'Schwimm'-Entscheidung mit diesen Befehlen doch mal um:

IF wetter = sonnig THEN
Packe Badezeug ein
Nimm Radio mit
Sonnenöl nicht vergessen
ELSEIF wetter = heiter
warten wir mal ab
ELSE
bleibe lieber zu Hause
lese die AMIGA DOS
END IF

Wir wollen also nur dann ins Freibad, wenn die Sonne lacht, gibt es bei Sonnenschein Bewölkung, warten wir lieber erst mal ab, ansonsten machen wir uns einen Nachmittag mit unserer Lieblingslektüre.

Schön und gut, nur möchten Sie gerne das alles in Aktion sehen, nicht wahr?

Listing 3 zeigt, wie man mit WHILE/WEND und IF/THEN/ELSE IF/ELSE/END IF wunderschön durchschaubare Programme erstellen kann, ohne auch nur ein einziges GOTO zu verwenden. Zwar erhebt es nicht den Anspruch auf der Weisheit letzten Schluß, aber gerade als Einsteiger kann man hieraus seinen Nutzen ziehen.

Gehen wir das Ganze doch einmal durch:

Außer dem Hauptprogramm mit dem Label PROGRAMM: befinden sich dort noch zwei Unterprogramme namens VARIABLE: und UNTERPROGRAMM:. In VARIABLE: wird der Variablen a der Tastenwert zugewiesen. Bei gedrückter Taste [1] wird a also 1, bei [2] wird a = 2, und so weiter.

In UNTERPROGRAMM: wird das Ergebnis auf den Bildschirm ausgegeben.

Im PROGRAMM: werden zunächst die benötigten Variablen deklariert, danach fragt eine WHILE-WEND-Schleife die Variable 'Flag1' ab. Solange wir das Erkennungsspielchen weitermachen wollen, ändert sich Flag1 nicht, erst bei der Antwort 'n', die durch UCASE$ in Großschrift umgewandelt wird, ändert sich Flag1, und das Programm wird beendet.

Bei der INPUT-Anweisung wird die Taste abgefragt, danach wird, je nachdem welche Taste gedrückt wurde, Flag2 mit einem der Situation entsprechenden Wert belegt. Wurden weder 1, 2 noch 3 gedrückt, geht das Programm in eine Abfrage, bei der man entscheiden kann, ob man weitermachen will oder nicht. Wie gesagt, das 'Nonplusultratotale' Programm ist dies nicht, es geht wesentlich einfacher, jedoch zeigt es, wie ich meine, die Funktion von IF und THEN und ihren 'Kollegen' ziemlich anschaulich.

Nachdem wir uns ausführlich mit dem Umgehen der meistbenutzten Fehler beim BASIC-Programmieren beschäftigt haben, wollen wir uns nun ein wenig mit SCREENS und WINDOWS beschäftigen.

Fenster auf – Fenster zu! Auch auf dem Amiga ist das möglich

In fast allen Beispielen, die wir Ihnen in den ersten beiden Folgen gezeigt haben, taucht eine Zeile auf, die Sie im ersten Moment zwar verwirrt hat, die Sie jedoch bestimmt kritiklos 'geschluckt' haben. Vielleicht hat Ihr Amiga aber auch mit einer Fehlermeldung gestreikt, die einer 'Guru-Meditation' nicht unähnlich sieht, sich jedoch ohne Absturz auswirkt. Diese Meldung 'Out of heap space' stellt sich vor allem auf A500ern ohne Speichererweiterung ein. Wenn Sie diese Meldung bekommen, dann schmeißen Sie doch einmal die Zeile aus dem Programm.

Und schon läuft es wieder, allerdings nicht mehr so schön wie vorher. Warum aber nicht?

Schauen wir uns die beiden Befehle genauer an, denn beide gehören ebenfalls zusammen wie Pech und Schwefel:

Ein Screen ist die Umschreibung für einen Bildschirm, dieser Screen zeigt uns alles, was vom Rechner auf den Monitor ausgegeben wird. Auf unserem Amiga-Monitor hat dieser Screen ein bestimmtes Seitenverhältnis, das sich aus der Breite und der Höhe der Pixel zusammensetzt. Ein Pixel ist ein einzelner Bildschirmpunkt, mehrere Pixel können Zeichen oder Grafiken bilden.

Der Befehl SCREEN öffnet unter BASIC einen Bildschirm, der bestimmte Eigenschaften hat. Die Syntax, also die Schreibweise eines solchen SCREEN-Befehls sieht dabei so aus:

SCREEN Nr,Breite,Höhe, Tiefe,Modus

Was bedeuten nun die einzelnen Eigenschaften, die wir als Parameter dem SCREEN-Befehl hinzufügen müssen?

Nr – Die Screen-Nummer. Unser Amiga kann bis zu vier separate Screens verwalten. Die Screen-Nummer wird später vom WINDOW-Befehl benötigt, um Windows im entsprechenden Screen zu setzen oder zu löschen.

Breite – Sie gibt an, wie groß unser neuer Screen von links nach rechts in Pixeln wird.

Höhe – steht sinngemäß für die Höhe der Screens.

Tiefe – Jetzt wird es schon schwieriger. Die Tiefe gibt an, wie viele Bit-Ebenen (Bitplanes) wir für den Screen haben wollen. Die Anzahl der Bitplanes ist entscheidend für die Anzahl der Farben, die wir verwenden wollen. Je mehr Farben, desto mehr Bit-Informationen müssen im Speicher vorhanden sein und desto mehr Bitplanes müssen zur Verfügung gestellt werden. Dabei ergibt sich folgendes Bild:

Tiefe	Anzahl der Farben
1	2 (0 und 1)
2	4 (Grundfarben0-3)
3	8 Farben
4	16 Farben
5	32 Farben

Ab einer Tiefe von 3 müssen die Farben vorher mit der *PALETTE*-Anweisung deklariert werden, doch dazu mehr im dritten Teil des Kurses.

Modus – Er ist ebenfalls wichtig, denn mit ihm geben wir die Größe des Screens an:

Modus	Screen
1	320 * 200
2	640 * 200
3	320 * 400
4	640 * 400

Die Screen-Angaben beziehen sich auf Breite x Höhe in Pixel.

Modus 3 und 4 erzeugen das sogenannte Zeilensprungverfahren, mehr bekannt unter dem Stichwort Interlace. Auch dazu kommen wir noch im nächsten Teil der Kurses.

Die Breite und die Höhe des Screens richten sich in dieser Form nach dem NTSC-Bildschirm, der vor allem in den USA-Amigas Verwendung findet. Unsere PAL-Monitore bieten da mehr Platz, genauer eine Auflösung von 640 x 256 Punkten. Experimentieren Sie ruhig einmal selber.

Der WINDOW-Befehl schließt sich an den SCREEN-Befehl an. Mit WINDOW legen wir den Bereich fest, indem wir einen Text ausgeben wollen. Wir müssen den WINDOW-Befehl benutzen, um den Text lesen zu können. Folgendes Programm enthält den Fehler, eine Ausgabe nicht sichtbar zu machen:

SCREEN 1,100,100,2,2
PRINT"Dieser Text ist nicht u sehen!!"
WHILE INKEY$ = "":WEND
SCREEN CLOSE 2:CLS:LIST

Nach dem Start verschwindet alles außer einer Screen-Titelleiste mit dem Front- und Hintergrundsymbol. Sonst nichts. 'Dieser Text ist nicht zu sehen!!' entspricht völlig der Wahrheit. Ziehen Sie jetzt doch mal mit dem Mauszeiger die Titelleiste herunter. Unser Text befindet sich im Output-Fenster und schlummert vor sich hin.

Bevor wir zur Syntax des Window-Befehls kommen, noch eine kurze Bemerkung: Windows brauchen wie die Screens eine Kennziffer, um den Text oder die Grafik an die richtige Adresse zu schicken. Die erste Kennziffer, die 1, ist reserviert für unser Output-Window, deshalb nehmen wir für alle neuen Ausgaben die Nummer 2.

Nun zur Syntax. Der Befehl lautet wie folgt:

WINDOW Nr,"Name",(xAnfang,yAnfang)-(xEnde,yEnde),Typ,Screen

Im einzelnen haben die Parameter folgende Bedeutung:

Nr – Die Window-Nummer, die benötigt wird, um alle Ausgaben an das richtige Window zu lenken.

"Name" – In der Titelleiste kann ein beliebiger Name, z.B. "Mein erstes Programm", eingesetzt werden.

(xAnfang,yAnfang)-(xEnde,y Ende) – Die Window-Begrenzung. Windows können natürlich nie größer sein als der Screen, innerhalb der Screen-Werte läßt sich aber alles einstellen. (xAnfang, yAnfang) bilden die linke obere Ecke, (xEnde,yEnde) die rechte untere.

Typ – Mit einem Window kann man eine Menge anstellen, diese Erfahrung haben wir auch schon mit der Workbench gemacht. **Typ** gibt an, was wir mit diesem Window machen wollen:

Typ = 1: Das Fenster kann mit der Maus durch das Größensymbol verändert werden.
Typ = 2: Das Fenster kann durch Anklicken der Titelleiste verschoben werden.
Typ = 4: Das Fenster enthält Hintergrund- und Vordergrundsymbol und kann nach vorne oder nach hinten gelegt werden.
Typ = 8: Das Fenster wird mit einem Schließsymbol ausgestattet.
Typ = 16: Der Inhalt des Fensters wird bei Überblendung eines anderen Windows zwischengespeichert und wieder angezeigt, wenn das Window wieder aktuell wird (Vorsicht, sehr speicherplatzfressend).

Die Werte können auch kombiniert werden, und zwar durch Addition. Geben wir dem Window Typ = 15, so enthält es die Eigenschaften von 1, 2, 4 und 8.

Screen – Er teilt das Window dem Screen mit der gleichen Nummer zu.

Ein kleines Programm mit etwas Grafik sehen Sie in Abbildung 4.

Zum WINDOW- und SCREEN-Befehl gehören noch Varianten: *WINDOW Nr* aktiviert das Window mit der Nummer Nr. Um die Ausgabe auf ein **nicht im Vordergrund befindliches** Fenster zu lenken benutzt man *WINDOW OUTPUT Nr*, damit kann man zum Beispiel Grafik im Hintergrund aufbauen, während im Vordergrund ein anderer Programmteil läuft. Dazu muß das Window jedoch mindestens Typ = 16 sein, um die enthaltenen Zeichen zwischenzuspeichern.

WINDOW CLOSE Nr

schließt das angegebene Window, *SCREEN CLOSE Nr*

```
Zuweisungen:
    x1=320:y1=128:r=0:Farbe=1
    SCREEN 2,640,256,4,2:'Screen 640*256 Pixel und 16 Farben
    WINDOW 2,"Kreise",,2,2:'Window fuer Screen 2 in voller Gro3e

Programm:
    WHILE INKEY$=""
        WHILE r < 128
            GOSUB Kreise
            r=r+1
        WEND
        WHILE r > 1
            GOSUB Kreise
            r=r-1
        WEND
    WEND
    WINDOW CLOSE 2:SCREEN CLOSE 2
    CLS:LIST:END

Kreise:
    CIRCLE (x1,y1),r,Farbe
    Farbe=Farbe+1:IF Farbe>15 THEN Farbe=0
    RETURN

Farben:
   PALETTE 4,1,1,1       'wei3
   PALETTE 5,1,1,.13     'gelb
   PALETTE 6,1,.73,0     'orange
   PALETTE 7,.93,.2,0    'helles rot
   PALETTE 8,.47,.87,1   'helles blau
   PALETTE 9,.73,1,0     'helles gruen
   PALETTE 10,.33,.87,0 'gruen
   PALETTE 11,0,.93,.87 'tuerkis
   PALETTE 12,1,.87,.73 'oker
   PALETTE 13,1,.13,.93 'violett
   PALETTE 14,.8,.6,.53 'braun
   PALETTE 15,.93,.53,.87 'Ueberraschung
   RETURN
```

Abb. 4: Mit dem SCREEN-und dem WINDOW-Befehl lassen sich neue Bildschirme mit bestimmten Eigenschaften öffnen

schließt den angegebenen Bildschirm.
Noch ein kleines Beispiel für SCREEN und WINDOW bildet Listing 5. Hier wird die Grafik erst gezeigt, wenn diese fertig ist.
Besitzer eines Amiga 500 mit 512 kByte werden allerdings wieder die ominöse Meldung 'Out of heap space' bekommen. Der Grund ist recht schnell erklärt: Window- und Screen-Operationen benötigen viel Speicher, vor allem dann, wenn Inhalte eines Windows zwischengespeichert werden müssen. Hier heißt es vorsichtig hantieren mit den Window- und Screen-Befehlen, nur wer über genügend Speicherplatz verfügt, kann uneingeschränkt 'fensterln'.
So, das war es für dieses Mal, im nächsten Heft gehen wir an eines der interessantesten Themen heran, die Grafik.

Haben Sie die Aufgabe gelöst? Nur Übung macht den (BASIC-) Meister

Unsere Aufgabe aus dem letzten Heft war ja wirklich einfach. Sie sollten mit dem LINE-Befehl eine Linie von oben nach unten auf dem Output-Window ziehen, die Farbe wechseln bis zu Farbe 3 und wieder zum Anfang zurück. Wenn Ihr Programm so aussieht wie dieses:

CLS
FOR Farben = 1 TO 3
FOR Anzahl = 1 TO 189
LINE (1,Anzahl)-(640,Anzahl),Farben
NEXT Anzahl
NEXT Farben

dann haben Sie es geschafft. Bei diesem Miniprogrämmchen kam es ja auch weniger auf die Programmierkünste, sondern mehr auf das Erlernen des Umgangs mit dem Editor an.

Unsere heutige Aufgabe soll das bisher Gelernte vertiefen:

Sicherlich kennen Sie noch diese kleinen Zahlenratespiele, die man im Deutschunterricht in der Schule so wunderbar hinter vorgehaltener Hand spielen konnte – jedenfalls so lange, bis der Lehrer mitspielte: Jemand denkt sich eine Zahl, und Sie müssen sie raten. Ist Ihre geratene Zahl zu groß oder zu klein, wird dies Ihnen gesagt, ist die Zahl gefunden, bekommen Sie die Anzahl der Versuche zu hören. In unserem Fall soll der Computer sich eine Zahl zwischen 1 und 1000 ausdenken, die Sie erraten müssen. Das Programm soll so lange ablaufen, bis die Zahl gefunden wurde, danach wird mit Anzeigen der benötigten Versuche geendet. Schreiben Sie sich ein solches Programm, aber mit folgenden Bedingungen:

Es darf kein GOTO benutzt werden! Greifen Sie auf IF/THEN/ELSE/ELSEIF/END IF zurück! Benutzen Sie WHILE/WEND!

Für diejenigen, die BASIC bis jetzt noch nicht kennen, ein paar Hinweise:

AmigaBASIC benutzt für Zufallszahlen die RANDOMIZE-Anweisung. Schauen Sie sich dazu im BASIC-Handbuch die Befehle RANDOMIZE und RND an. Für unseren Zweck lohnt sich die Funktion RANDOMIZE TIMER, die Zufallszahlen unter Zuhilfenahme der internen Timerbausteine erzeugt.

So, das war er, der zweite BASIC-Kurs-Teil. Nachfolgend finden Sie ein Glossar, indem die im Text vorkommenden AmigaBASIC-Befehle noch einmal erklärt werden.

Bis zum nächsten Mal!

(jb)

```
REM Blick ins Weltall
   SCREEN 2,640,256,2,2
   WINDOW 2,"Grafik",,31,2 'Hier wird die Grafik erstellt!
   WINDOW 3,"Text",,31,2   'Hier wird vorher Text ausgegeben!
   WINDOW OUTPUT 3
   PRINT"Bitte etwas Geduld!!" 'Time is money!!
   WINDOW OUTPUT 2             'Die Grafik wird erstellt.
   FOR n=1 TO 640 STEP .5
      LINE(320+100*COS(n/64),128+100*SIN(n/64))-(320+100*SIN(n/8),128+100*COS(n/8)),1
   NEXT n
   WINDOW 2                 'Die Grafik ist fertig!!
   WHILE INKEY$="":WEND     'Wenn's reicht, Taste druecken!!
   WINDOW CLOSE 2           'Es wird alles geschlossen!
   WINDOW CLOSE 3
   SCREEN CLOSE 2
   END
```

Abb.5: Auch versteckte Abläufe sind durch geschicktes Programmieren nötig

Festplatten

28 ms, OMTI - Controller, komplett für AMIGA 500/1000, **AutoPark,** durchgeführter Expansionsport, Boil! - Treiber, Einbaumöglichkeit einer 2. Platte, sehr schnell, **1 Jahr Garantie,** Gehäuse als Monitoruntersatz verwendbar, ca 32x32x6 cm Bei Drucklegung standen die genauen technischen Daten noch nicht fest. Bitte fordern Sie unverbindlich und kostenlos unsere aktuelle Produkt- und Preisübersicht an.

42 MB 400 KB/S 28 ms **1248.-**

42 MB 24 ms, 400 KB/S, sehr leise **1298.-**

65 MB 28 ms, 330 KB/S **1398.-**

122 MB 28 ms, 330 KB/S **2098.-**

Preisermäßigung für AMIGA 2000 DM 150.-

Testauszug AMIGA (Ausgabe 9/89, Seite 151) :
"Als Treibersoftware kommt BOIL (Bootable Omti Interface Loader) zum Einsatz. **BOIL ist einer der schnellsten Festplattentreiber, die für den AMIGA verfügbar sind.** Bei Verwendung einer Seagate ST 251 mit einem OMTI 5520 erreichte die AMIGOS Festplatte eine Schreib/Lese Geschwindigkeit von 380/220 KByte/s. BOIL arbeitet mit allen Festplatten zusammen, die einen OMTI - Controller verwenden. Ab Kickstart 1.3 ist die Festplatte FastFileSystem bootfähig. Auf der mitgelieferten Systemdiskette befinden sich Hilfsprogramme zum Formatieren, ein Festplattentest und ein Programm, daß das Interface auf Funktionstüchtigkeit untersucht."

Testauszug Kickstart (Ausgabe 10/89, Seite 19) :
"Die mit dem BOIL - Treiber installierten Platten sind rebootable unter Kickstart 1.3, d.h. daß nur einmal von Diskette gebootet werden muß. Nach einem Reset wird dann automatisch von Festplatte gebootet, wobei auch die Bootpartition unter FFS gewählt werden kann. Die Arbeitsgeräusche der Platten sind gering, auch der leise Lüfter trägt dazu bei. **Insgesamt machen die Festplatten von Frank Strauß Elektronik einen sehr guten Eindruck, was nicht nur an dem hervorragenden BOIL Treiber liegt, sondern am ganzen Konzept;"**

Diskettenlaufwerke

Vollkompatibel, anschlußfertig, abschaltbar, inkl. Kabel, amigafarbenem Metallgehäuse, 2x80 Spuren, alle Laufwerke mit beiger Frontblende. Wir verwenden nur Markenlaufwerke von **TEAC** (FD 235 F oder FD 55 (G)FR) und **NEC** (1037a). Alle 5.25" Stationen werden mit **40/80** Trackumschaltung, **durchgeführtem Bus und original Commodore - Treiberplatine** geliefert. Auf alle **TEAC** Diskettenlaufwerke gewähren wir **1 Jahr Garantie.** Durchgeführter Bus (3.5"): DM +10.-

NEC **TEAC**

3.5": 229.- **5.25": 259.-**

Festplattentreiber

BOIL! = 400 KB/S
Bootable OMTI Interface Loader

DM 75.-

Frank Strauß Elektronik
Schmiedstraße 11
6750 Kaiserslautern
Tel.: (0631) 67096 - 98
Fax: 60697
Lieferung erfolgt ab Lager mit UPS oder Post per Nachnahme.
Donnerstags bis 20.30 h geöffnet.

BASIC-Glossar, Teil 1

In unserem BASIC-Glossar finden Sie außer den Befehlen, die im Text beschrieben werden, auch artverwandte Befehle:

CHR$ (ASCII)
Die Funktion CHR$(ASCII-Wert) gibt das Zeichen 'ASCII-Wert' aus. Beispiel: PRINT CHR$(65) gibt auf dem Bildschirm das große 'A' aus.

COLOR (Hintergrundfarbe, Vordergrundfarbe)
Mit COLOR werden Hintergrund- und Vordergrundfarbe festgelegt. Beispiel: COLOR 1,0. Der Amiga verfügt beim Start nur über die vier Grundfarben, nach Einsatz des SCREEN-Befehls lassen sich mit der PALETTE-Anweisung bis zu 32 Farben erstellen (siehe auch PALETTE).

IF...THEN (ELSEIF, ELSE, END IF)
Befehle zur bedingten Verzweigung. Die IF-Bedingung ist dann erfüllt, wenn sie nach Überprüfung als WAHR gesetzt wurde. IF X = 1 kann dabei ebenso die Bedingung WAHR erfüllen wie X< >1 (X ungleich 1). Mit ELSEIF können weitere Bedingungen angeschlossen werden, die jedoch im Gegensatz zur ersten stehen. ELSE ist die Alternativ-Bedingung. END IF schließt eine Reihe von Anweisungen nach IF...THEN ab.

INSTR (Startposition, Stringvariable, Zeichenkette)
Mit INSTR wird in einer Stringvariablen eine Zeichenkette gesucht und deren Position ausgegeben. 'Startposition' wird nicht unbedingt benötigt, gibt aber an, ab welcher Position gesucht werden soll. Beispiele:

a$ = "SuchString":PRINT INSTR(a$,"S")

gibt eine 1 aus, da nur das erste 'S' gesucht wird.

a$ = "SuchString":PRINT INSTR(2,a$,"S")

gibt eine 5 aus, da erst ab Position 2 gesucht wird.

INKEY$
übergibt ein Zeichen aus dem Tastaturpuffer. *w$* = INKEY$ hat zur Folge, daß 'w$' das zuletzt eingegebene Zeichen enthält. Kleiner Tip: Will man in einem Programm so lange warten, bis eine Taste gedrückt wurde, empfiehlt sich folgende Zeile:

WHILE INKEY$ = "":WEND

(siehe auch unter WHILE/WEND).

INT (Variable)
Die Funktion INT (von INTeger = Ganzzahl) macht aus einer Fließkommazahl einen ganzzahligen Ausdruck. Beispiel: PRINT INT(13.25) ergibt 13. Dies wird dann genutzt, wenn man nicht mit Fließkommazahlen arbeiten möchte oder kann. Für den LINE-Befehl sind zum Beispiel nur Integerzahlen interessant, da die Pixelpunkte feste Koordinaten haben (0 bis 640 oder 0 bis 256). VORSICHT! INT rundet Zahlen nicht auf oder ab, sondern schneidet den Nachkommateil rigoros ab (siehe auch unter RND/RANDOMIZE).

LEN (Stringvariable)
Die Funktion LEN gibt die Länge einer Stringvariablen aus. Stringvariable sind zum Beispiel Namen oder Kombinationen aus Zeichen wie "Hallo Welt", "Wie lautet Ihr Name?", "Listing 1" und ähnlichem.

a$ = "Hallo Welt":PRINT LEN(a$)

ergibt als Ergebnis 10. Zählen Sie nach!

LEFT$ (Stringvariable)
Die Funktion LEFT$ gibt eine gewisse Anzahl an Zeichen einer Stringvariablen aus, die vom ersten Zeichen aus gezählt werden. Beispiel:

a$ = "Linke Zeichen":PRINT LEFT$(a$,6)

Man erhält als Ergebnis "Linke ", das Leerzeichen wird als Nummer 7 mitgezählt (siehe auch RIGHT$ und MID$).

MID$ (Stringvariable)
Mit MID$ kann ein Teil eines Strings ausgelesen werden, wobei Länge und Position des Teilstrings angegeben werden müssen. Beispiel:

a$ = "Zeichen in der Mitte":PRINT MID$(a$,9,6)

Ausgegeben wird "in der", angefangen beim neunten Zeichen, die Länge geht bis Zeichen 14. Überprüfen Sie es! (Siehe auch unter LEFT$ und RIGHT$.)

RANDOMIZE (Anfangswert/TIMER)
Der Befehl RANDOMIZE aktiviert den Zufallsgenerator. Benötigt wird als Parameter der Anfangswert, der im Bereich von -32768 bis 32767 liegen kann. Gibt man RANDOMIZE ohne Parameter an, fragt AmigaBASIC selbständig danach.
Erwarten Sie jedoch lieber keine richtigen Zufallszahlen: Nach einiger Zeit wiederholen sich alle ausgegebenen Werte. Besser ist die Funktion RANDOMIZE TIMER, hier werden die Timerbausteine abgefragt, und zwar im Bereich von 1/100 Sekunden. Die daraus resultierenden Zahlen liegen zwar alle kleiner 1, geben aber bessere Zufallszahlen ab. Um jetzt an konkrete Zufallszahlen zu kommen, ist ein kleiner Gedankensprung nötig: Wir erhalten Zahlen, die kleiner 1 sind, wollen aber Zahlen zwischen 0 und 50 haben (also 1 bis 49, Lotto läßt grüßen). Dann benötigen wir folgende Zeilen:

RANDOMIZE TIMER
*Zufallszahl = INT(RND(49)*49) + 1*

Die mit RND erzeugte Zufallszahl (im Bereich von 0 bis 1) wird mit 49 multipliziert, dazu wird eine 1 addiert, um die Null auszufiltern, das Ganze wird jetzt noch mit INT auf ganze Zahl getrimmt (siehe auch INT und RND).

RIGHT$ (Stringvariable)
RIGHT$ ist das genaue Gegenteil von LEFT$, hier wird der rechte Teil einer Stringvariablen ausgelesen. Beispiel:

a$ = "Rechte
Zeichen":PRINT
RIGHT$(a$,7)

Als Ergebnis bekommt man "Zeichen", da von rechts nach links gezählt wird. Auch hier gilt: Ausprobieren! (Siehe auch unter LEFT$ und MID$.)

RND (Wert)
erzeugt Zufallszahlen zwischen 1 und 0. Um an 'vernünftige' Werte zu kommen, ist RND nur sinnvoll im Einsatz mit RANDOMIZE (siehe auch RANDOMIZE und INT).

SCREEN Nr, Breite, Höhe, Tiefe, Modus
Der SCREEN-Befehl öffnet einen neuen Bildschirm (nicht wörtlich zu nehmen) unter AmigaBASIC. Dieser Bildschirm kann verschiedene Größen und Auflösungen sowie eine unterschiedlich große Anzahl Farben enthalten. Beispiel:

SCREEN 2,640,256,2,2

öffnet Screen 2 mit einer Breite von 640 × 256 Pixeln (Breite × Höhe), einer Tiefe von 2 (4 Farben möglich) und dem Modus 2, der die Größe von 640x256 festlegt. Im Anschluß an den SCREEN-Befehl muß der WINDOW-Befehl kommen. Screens können mit SCREEN CLOSE Nr geschlossen werden (siehe auch WINDOW).

UCASE$ (Stringvariable)
macht aus Kleinbuchstaben innerhalb einer Stringvariablen Großbuchstaben. Beispiel:

a$ = "abc":PRINT UCASE$(a$)

Als Ergebnis erhalten wir "ABC".

WINDOW Nr,"Name",(x1,y1)-(x2,y2),Typ,Screen
WINDOW erstellt Fenster auf einem vorher definierten Screen. Beispiel:

SCREEN 2,640,256,2,2
WINDOW 2,"Mein Window"○1,2

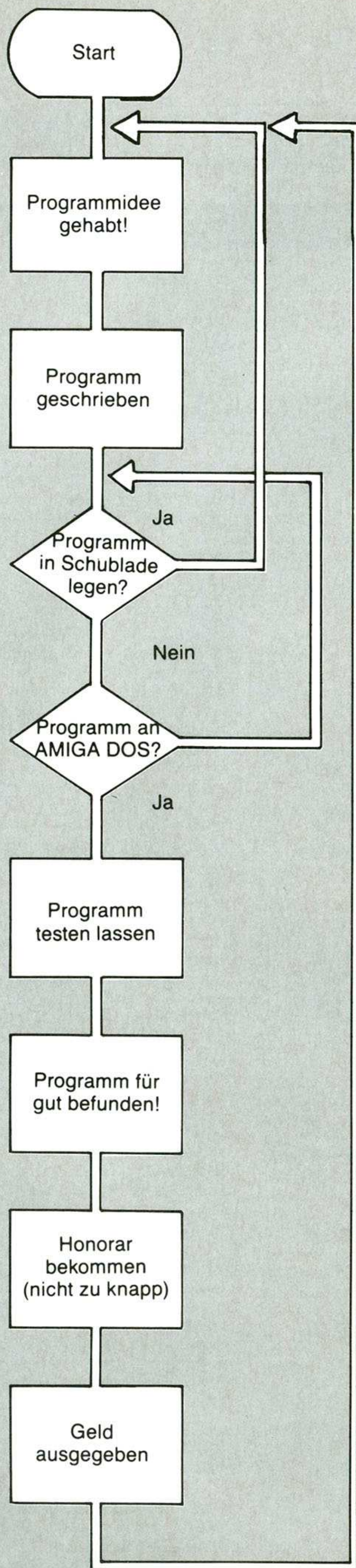
öffnet ein Window im Screen 2 mit der Kennung "Mein Window", der vollen Größe von 640x256 Pixeln (daher keine Angabe notwendig), des Typs 31, mit allen Veränderungsvariationen (siehe Text) und der Screenzugehörigkeit 2 (zu Screen 2). Windows können angewählt werden mit WINDOW Nr, die Ausgabe auf ein Hintergrund-Window mit WINDOW OUTPUT Nr gesetzt werden, oder sie können mit WINDOW CLOSE Nr geschlossen werden.

WHILE/WEND
dienen zur Schleifenprogrammierung. Beispiel:

WHILE a>1
PRINT a
a = a-1
WEND

Das Programm zeigt die Variable 'a' solange an, bis diese den Wert 1 erreicht.

VERSTEHEN SIE DAS?



Programmieren Sie für die AMIGA DOS!!

Haben Sie ein Super-Utility, die ultimative Dateiverwaltung oder den tiefblickendsten Diskettenmonitor für den Commodore Amiga geschrieben? Liegt Ihr Meisterwerk nun unbekannt und ungebraucht in der Schublade? Dann haben wir einen Vorschlag für Sie! **Senden Sie Ihr bestes Programm doch einmal an die Redaktion der AMIGA DOS ein.** Denn wenn uns Ihr Programm ebenso gut gefällt wie Ihnen, dann können Sie es kurz darauf in der AMIGA DOS gedruckt wiederfinden. **Gegen ein entsprechendes Honorar, versteht sich!** Und wenn uns Ihr Programm sogar noch besser gefällt als Ihnen selbst, dann hat es die gute Chance, **Programm des Monats** zu werden, eine Auszeichnung, die nur exzellenten Programmen zuteil wird. Also ran an die Maschine und programmiert. Oder ran an die Schublade und nachgeschaut. Es lohnt sich! Bestimmt!!

Ach ja– Einsenden können Sie es an den

DMV- Verlag
Redaktion AMIGA DOS
Postfach 250
3440 Eschwege

Auf geht's!!

Edgar Meyzis

Aufsteigerwissen Assembler

(Teil 2)

Hoffentlich hat der erste Teil des Kurses nicht mehr Fragen aufgeworfen als beantwortet? Selbst wenn das so sein sollte, bleiben Sie dran. Wir liefern heute die zugesagten Erläuterungen zur "Paint Box", um eventuell noch vorhandene Lücken zu schließen. Weiterhin konstruieren wir unter anderem eine multitasking-gerechte Programmsteuerung und bringen es unserer Paint-Box bei, resident-ladbar zu sein. Als Schmankerl packen wir die Eigenschaft "reentrant" noch oben drauf. Wo gibt es das schon?

Kursfahrplan
Teil 1. Einführung und Grundlagen,
Teil 2. Vertiefung der Programmiertechnik
Teil 3. Praktische Arbeit mit Graphics
Teil 4. Praktische Arbeit mit Dos
Teil 5. Hardwarenahe Programmierung von Grafik

Paint Box 1.0

Bevor wir uns an den neuen Stoff heranmachen, soll das Listing 4 der ersten Folge gründlich besprochen werden, um auch zu "Späteinsteigern" (Per Anhalter durch die Wüste?) nett zu sein. Das Programm besteht aus den Teilen:

- **Offset-Vektoren,**
- **Makros,**
- **Unterprogramme,**
- **Hauptprogramm und**
- **globale Daten.**

Der vom Assembler generierte Code beginnt mit einem Sprung über die Unterprogramme hinweg zum Hauptprogramm (Main). Hier hat sich ein Fehler eingeschlichen. Die Anweisung "jsr Main" ist durch "bra Main" zu ersetzen. Sorry!
Für die umfangreichen Makros wird kein Code generiert. Die Makros sind lediglich Textbausteine, die bei der Assemblierung in den Programmtext anstelle der Makroaufrufe eingesetzt werden.
Im Hauptprogramm, Wüstenfüchse wissen es schon, werden zunächst die Bibliotheken geöffnet. Das Makro "LibsOeffnen", das wiederum auf "OpenLib" zugreift und zugleich Unterprogramm ist, besorgt diese Aufgabe für uns. Die Adressen der Bibliotheken werden in "GraphLib" und in "IntuiLib" abgelegt. Sollte eine Bibliothek nicht geöffnet werden können, so enthält d0 den Wert null, und mit "beq endeL" wird zu "LibsSchliess" verzweigt, um geöffnete Bibliotheken zu schließen.
Bei uns geht ja alles glatt, und deshalb wird in der Folge das Unterprogramm "ScreenAnlegen" aufgerufen. Die Routine nutzt die vorher geöffnete Bibliothek "Intuition" und den Offset-Vektor "OpenScreen" Die Adresse des Screen ist in d0 enthalten. Sie wird in der globalen Variable "Screen" abgelegt. Wäre der Systemaufruf "OpenScreen" erfolglos, so würde sich Null im Register d0 befinden. Wir können somit wieder auf Null testen, um notfalls zu "endeS" zu verzweigen. Warum zu "endeS"? Wenn kein Screen angelegt wurde, dann gibt es auch nichts abzubauen. Richtig! Sichere Programme sollten so robust sein, daß sie möglichst mit allen Eventualitäten fertig werden. Die Routine "ScreenSchliess" wurde deshalb auch so angelegt, daß zum Ende verzweigt wird (Label "\s"), wenn eine Adresse ungleich Null in "Screen" enthalten ist. Das galt es zu demonstrieren.
Erkennen Sie nun die Schachtelung der Aufrufe von Systemroutinen? Sicher doch! Die äußere Schicht bilden die Aufrufe "LibsOeffnen" und "LibsSchliess", die mittlere besteht aus "ScreenAnlegen" bzw. "ScreenSchliess" und die innere aus "WindowAnlegen" bzw. "WindowSchliess". Entsprechend dieses Beispiels ist stets zu verfahren, wenn Systemmittel eingesetzt werden sollen, in diesem Fall Arbeitsspeicher für Screen und Window.
Zurück zu unserem Screen. Vor dem Öffnen ist er zu definieren. Der einzuhaltene Rahmen und die Möglichkeiten sind gut in [1] erläutert. Die "NewScreenData" enthalten die Beschreibung, die viel ausführlicher hätte ausfallen sollen, sofern der Platz dazu gereicht hätte. Wir kommen darauf noch einmal zurück, wenn wir an der nächsten Version der "Paint Box" (Listing 3 dieser Folge) arbeiten.
Die Ausführungen zum Screen gelten im übertragenen Sinne auch für das Anlegen des Window. Mit der Anweisung "move.l d0,nw__screen" tragen wir die Adresse des gerade geöffneten Screen in die Beschreibung des Window (NewWindowData) ein, um anschließend das Fenster zu öffnen. Natürlich klappt auch das, und wir haben die Bühne für unsere Grafikexperimente geschaffen.
Die Bühne bliebe leer, und wir müßten auf eine Fata Morgana hoffen, wenn wir nicht die beiden Unterprogramme "FuelleEi" und "ZickZack" hätten. Beide verhalten sich in etwa namensgerecht. Wir halten es schon für wichtig, möglichst anschauliche Bezeichner zu verwenden, wie Sie den bisherigen Ausführungen entnehmen konnten. Der Assembler "A68k" unterstützt dabei, indem praktisch beliebig lange Namen zugelassen sind. "FuelleEi" setzt eine Reihe von Routinen ein, die als Makros formuliert sind. Damit sollte gezeigt werden, daß es mit Makros leicht möglich ist, eine sprachliche Oberfläche für Assembler zu schaffen, die Hochsprachen ähnelt. Vielleicht konnte damit auch eventuell vorhandene Scheu vor den Unbilden einer Wüstendurchquerung abgebaut werden. Im Umgang mit Makros sind wir nun sicher. Haben Sie sich auch darüber gewundert, daß die "Flood-Routine" soviel Zeit beansprucht?
Die Routine "ZickZack" hatte unter anderem die Aufgabe, den globalen Gebrauch von Registern zu zeigen (XPos = d6 u. Farbe = d5). Variablen in Registern beschleunigen Programmabläufe. Als Programmierer muß man dann höllisch aufpassen, die Registerinhalte nicht zu zerstören. Die Routine sollte aber auch Farben auf den Bildschirm bringen, und zwar alle, die in den ersten 16 Farbregistern enthalten sind. Unser Screen hat vier Bit-Planes, kann somit "zwei hoch vier" Farben darstellen. Wir beginnen deshalb auch mit dem Farbregister "15", zählen die Registernummer herunter und beginnen wieder von vorne. Für die Farbangabe werden nur die unteren vier Bits von "Farbe" benötigt. Es kommt somit zu einem automatischen "wrap around". Auf Null folgt 15. Ging das zu schnell? Dann malen Sie doch bitte die Bits in den Wüstensand, um den Sachverhalt nachzuvollziehen. (Zur Not tut es auch ein Stück Papier.) Es ist wichtig! Übrigens haben wir mit "ZickZack" auch gleich unsere Kür in rekursiver Programmierung absolviert. Der letzte Teil der Routine (beginnend ab "Zick") ruft sich wiederholt selbst auf.
Bleibt uns noch die Beantwortung der Fragen aus der letzten Folge, bevor wir den Motor für die nächste Etappe anwerfen. Die Anworten finden Sie in Tabelle 1. Selbst wenn Sie daran nicht sonderlich interessiert sind, sollten Sie einen Blick zumindest auf Tabelle 2 werfen. Wir haben die Farben aufgeführt, die standardmäßig in den unteren 16 Farbregistern enthalten sind.

Erst einmal aufräumen..

Und nun freie Fahrt für die zweite Wüstenetappe. Wenn man unser Wüstenschiff eh.. "Paint Box 0.1" betrachtet,

dann hat zwar jedes Stück Code seinen festen Platz, muß aber alles offen herumliegen? Wie wäre es, wenn wir die weniger häufig benötigten Teile wegpackten, um an Übersichtlichkeit zu gewinnen? (Wie resistent sind eigentlich Bits gegen Wüstensand?) Gesagt, getan. Wir legen drei neue Dateien im Unterverzeichnis "incl" innerhalb unseres Projektverzeichnisses an. Dazu kopieren wir die Aufstellungen mit den Offset-Vektoren (Folge 1, Listing 4) in die Dateien "incl/Exec.i", "incl/Intuition.i" und "incl/Graphics.i" und ersetzen die Passagen im ursprünglichen Quelltext durch die Anweisungen "include "incl/Exec.i" usw. In "Exec.i" sollten Sie außerdem "execBase" sowie die Makros "CallExec", "OpenLib" und "CloseLib" übertragen. Damit haben wir den ersten Schritt vollbracht, die Programmiertechnik fahrplangemäß zu verbessern. Das Programm, wir nennen es ab jetzt "P2", läßt sich wie bisher assemblieren.

Wir können aber noch weitere Programmteile auslagern, z.B. die Routinen "Fuelle Ei" und "ZickZack", die dazugehörigen Makros sowie die Registerdefinitionen "XPos" und "Farbe". Die erste Routine könnten Sie auch über Bord werfen, da nicht weiter benötigt. Wir haben es getan, zumal es in unserer Wüste keinen Umweltschutz gibt. (Na, na, so schlecht war das Spiel mit den Makros doch gar nicht.) "ZickZack" wandert aber nicht in das Unterverzeichnis "incl", sondern wird unter dem Namen "G2.asm" (kurz G2) zusammen mit den von dort aus aufgerufenen Makros im selben Verzeichnis versteckt, das auch "P2.asm" beherbergt.

Sie könnten es nun versuchen, "G2" zu assemblieren. Es würde nicht klappen! Warum eigentlich nicht? Kratz, kratz..., in "Grafik0" sind die gerade angelegten Include-Dateien unbekannt, und außerdem greift "ZickZack" auf unser Window sowie auf "GraphBase" zu, die aber bisher nur in "P2" erwähnt sind. Die Lösung zu Problem eins kennen Sie bereits. Das Problemchen zwei haben wir schneller gelöst, als einen Radwechsel in praller Sonne. Dazu teilen wir "ZickZack" vor Aufruf der Routine im Quelltext durch "PUBLIC Window" und "PUBLIC GraphBase" einfach mit, daß die Adressen beider Objekte erst während des Bindens zur Verfügung gestellt werden. Spätestens jetzt wird klar, wie Linker zu ihrer Bezeichnung gekommen sind.

"G2" läßt sich nun assemblieren, beim Binden grüßen uns jedoch Fehlermeldungen. "P2", das den Status eines Hauptprogramms hat, muß dem Linker gleichfalls "Window" und "GraphBase" zur Verfügung stellen, damit der Knoten gebunden werden kann. Dazu fügen in "P2" die Zeilen "PUBLIC Window" bzw. "PUBLIC GraphBase" ein. Selbstverständlich ist "P2" neu zu assemblieren. Nanu, schon wieder ein Fehler? Die Lösung finden Sie in Tabelle 3, Punkt a. Zum Binden geben wir ein:

"BLINK obj/P2.o obj/G2.o TO bin/P2"

Das Hauptprogramm ist zuerst zu nennen. Hat's geklappt? Falls nicht, so gehen Sie bitte den vorherigen Absatz noch einmal durch. Hoffentlich haben Sie noch eine Kopie der letzten lauffähigen Version. Tja, auch im kleinen sollte man eine saubere Konfigurationskontrolle ausüben. Schon ein Sandkorn (meine Fehler) kann viel Zeit verschlingen, wenn die Ausgangsbasis unsicher ist.

Viel haben wir bis jetzt noch nicht geleistet, außer "FuelleEi" über Bord zu werfen, könnte man meinen. In Wirklichkeit haben wir wichtige Schritte unternommen, unseren Programmierstil zu entwickeln. Das kleine Projekt hat weiter an Struktur gewonnen. Wir müssen aber noch mehr in die reine Programmiertechnik investieren, um unseren Kursfahrplan einzuhalten und so Ihren Erwartungen zu entsprechen.

Auch in der Assemblerprogrammierung hat es sich bewährt, Variablen zu Datenstrukturen zusammenzufassen und einzelne Elemente mit Namen anzusprechen.

Struktur, Verbund, wie Sie es mögen

In "C" bezeichnet man solche Objekte als "STRUCTURE" und in "Modula-2" als "RECORD". Wir wollen diesen Weg auch gehen, um die Übersichtlichkeit und Lesbarkeit unserer Programme zu erhöhen. Dazu legen wir die Datei "incl/Types.i" gemäß Listing 1 an. Die darin enthaltenen Makros dienen dazu, Offsets für einzelne Elemente von Datenstrukturen festzulegen und den dafür benötigten Speicherplatz zu ermitteln. Über die Offsets, die Namen tragen, ist es möglich, Elemente leicht nachvollziehbar zu adressieren (Adreßregister indirekt mit Offset).

Angenommen, wir wollen drei Variablen unterschiedlicher "Breite" zu einer Struktur zusammenfassen, Lang (Langwort), Kurz (Byte) und Mittel (Wort). Auf der Basis von Listing 1 deklarieren wir:

```
Test
struktur makro
long ts_Lang
byte ts_Kurz
byte ts_dummy
word ts_Wort
struktLaenge ts_size
```

Der Strukturname ("Test") ist für unsere Zwecke bedeutungslos. Das Makro "struktur" setzt lediglich einen internen Zeiger auf Null. Welcher mysteriöse interne Zeiger ist damit gemeint? Welchen Namen trägt er? (Antwort siehe Tabelle 3, Punkt e.) Der interne Zeiger enthält stets die aktuelle Länge der Struktur.

```
a: Makros sind Textbausteine. Sie werden bei der Assem-
   blierung an der Stelle ihres Aufrufes in den Quelltext
   eingefügt und erzeugen dann Code.
b: CallGraph macro
          move.l GraphBase,a6
          jsr Ö1(a6)
          endm
c: und d: siehe Tabelle 2
e: Die Ellipse wird am Fensterrand ohne Nebenwirkungen sau-
   ber abgeschnitten. Dieser "Service" wird nur in einem
   Window geboten. Auch ein Screen verfügt über einen Rast-
   port, der anstelle des Window-Rastport hätte eingesetzt
   werden können. Dadurch wäre an Ausführungsgeschwindigkeit
   zu gewinnen. Das "Clippen" an den Kanten des Screen blie-
   be dann jedoch von uns selbst zu erledigen, um andere
   Speicherinhalte nicht zu zerstören.
f: "RectFill" ist deutlich schneller.
g: Die Spitzen werden abgeschnitten (siehe e:).
h: Die Linie wird bis an den unteren Fensterrand gezogen,
   mehr nicht.
```

Tabelle 1

Unser Makro "long" trägt "ts_Lang" als Label ein, dem der Wert null zugewiesen wird. Das erste Element hat keinen Versatz. Der interne Zeiger wird auf vier erhöht.

Als nächstes trägt das Makro "byte" das Label "ts_Kurz" ein, dem der aktuelle Wert des internen Zeigers zugewiesen wird, das heißt vier. Alles klar? Nein, dann müssen wir noch auf "ts_dummy" eingehen. Die CPU kann die nächste Variable ("ts_Wort") nur adressieren, wenn sie sich auf einer geraden Adresse befindet. Es war notwendig, "dummy" einzufügen, um die Ausrichtung zu erreichen und einen korrekten Offset bzw. die exakte Länge der Struktur zu erhalten.

Fenster mit Struktur

Unser gerade entwickeltes Werkzeug zur Bildung von Datenstrukturen haben wir in einer neuen Version von "Intuition.i" (Listing 2) angewandt. Sie haben sicherlich erkannt, daß längere Strukturen spaltenweise definiert sind. Die Spalten sind (vor dem ersten Gebrauch von "Intuition.i") aneinanderzureihen. Mühselige Arbeit, die wir uns zunächst für die Struktur "Screen" schenken, dafür aber "sc_viewPort = $2C" deklarieren. Wenn sie nun "P2" assemblieren, dann werden Ihnen viele Fehler entgegenlachen. Warum, seit wann werden in der Wüste Nägel ausgestreut? Die Antwort finden Sie in Tabelle 3, Punkt b. So, nun sollte auch dieser Schritt erfolgreich vollzogen werden können.

Resident und reentrant? Ja bitte!

Auf unserer Fahrt durch die Wüste erreichen wir nun das Tal namens "Re-re". Es ist tückisch zu befahren. Ein Ausläufer in Form einer tiefen Spur wird uns die gesamte Zeit begleiten. Wir wollen hindurch und steigen auf ein geländegängigeres Fahrzeug um, auf die neue Fassung des Hauptprogramms, "PaintBox 0.2", wie in Listing 3 (kurz L3) dargestellt. Zunächst benötigen wir eine Datenstruktur, die unsere globalen Variablen aufnimmt, wie am Anfang des Listings dargestellt. Um die bereits erwähnte, indirekte Adressierung innerhalb einer solchen Struktur zu unterstützen, definieren wir "globals

equr a5" und beschließen, das Register a5 exklusiv als Zeiger auf die Globalen zu reservieren (L3, unter Registergebrauch).

Die Struktur ist nun beschrieben. Es fehlt uns nur noch der Arbeitsspeicher zur Aufnahme der Variablen. Abweichend von Folge 1 fordern wir am Anfang von "Main" mit dem Makro "AllocMemory" entsprechenden Speicherplatz an und geben ihn unmittelbar vor Beendigung des Programms wieder frei. Vorher sind jedoch noch "Exec.i" entsprechend Listing 4 und "Graphics.i" entsprechend Listing 5 zu ergänzen.

Damit haben wir einen ersten wichtigen Schritt zur Entwicklung eines "residentfähigen" Programms vollzogen. Es muß sichergestellt sein, daß globale Variablen bei Programmstart stets einen bekannten Zustand aufweisen. Das erreichen wir z.B. durch Anforderung von Arbeitsspeicher zur Aufnahme der Globalen, den wir zugleich mit Null initialisieren.

Aber auch in Richtung "Reentrancy" (Wiedereintrittsfähigkeit) haben wir uns bewegt. Durch die Anforderung von Speicher wird bei jedem Programmaufstart (desselben Codes) ein eigener Satz globaler Variablen angelegt. Ohne einen derartigen Schritt wäre Code nicht "reentrant" zu schreiben (siehe dazu ausführlich [2]).

Von den Variablen "GraphBase", "IntuiBase", "Screen" und "Window" haben wir uns getrennt, wie ein Blick auf den Schluß von L3 zeigt. In "P2" und in "G2" tritt an die Stelle von "Screen" "Screen(globals)". "Window" ist durch "our_Win" zu ersetzen (siehe L3, Registergebrauch). Und wie sprechen wir die Basisadressen der Bibliotheken an? Die Antwort, wie immer, finden Sie in Tabelle 3, nun unter Punkt c. Der Texteditor kann bei den vorzunehmenden Änderungen gute Dienste leisten.

Sie werden erkennen, daß "globals" und "our_Win" nicht mit "PUBLIC"-externen Programmteilen bekannt gemacht werden können. No problem! Wir deklarieren in "G2" die Registervariablen ebenfalls. Ein Namenskonflikt tritt nicht ein. Der Anfang von "G2" sieht dann gemäß Listing 6 aus.

Nun könnte die Frage aufkommen, warum die mit "GraphLib", "IntuiLib" und "ScreenTitle" bezeichneten Objekte nicht Eingang in unsere Struktur der Globalen Eingang gefunden haben? (Tabelle 3, Punkt d)

Die "NewScreenData" und die "NewWindowData" wollen wir Ihnen nicht unterschlagen. Wir ersetzen sie durch die Routinen "ScreenAnlegen" und "WindowAnlegen" gemäß L3. In den Routinen fordern wir (mit Null initialisierten) Arbeitsspeicher im Umfang der jeweils benötigten Struktur an, initialisieren die enthaltenen Variablen entsprechend, öffnen den Screen bzw. das Window und geben anschließend den Speicher wieder frei.

Multitasking-gerechte Programmsteuerung

Stürzen wir uns auf ein weiteres Zwischenziel, die Programmsteuerung, die anfangs nur unsere hardwarenahe Abfrage des linken Mausknopfes ersetzen soll. Dazu dient die Routine "ProgramControl" in "P2" (L3) unmittelbar vor "Main". Der Aufruf "jsr Taste" (Folge 1) ist durch "bsr Programcontrol" ersetzt. Das "IDCMP-Flag" in der "NewWindow-Struktur" (Routine "WindowAnlegen") veranlaßt Intuition, unser Programm über die Betätigung der linken Maustaste zu informieren.

Was leistet "ProgramControl"? Zunächst wird geprüft, ob am "UserPort" des Window eine Nachricht vorliegt (mit "GetMsg" und "tst.l"), die z.B. durch Drücken des linken Mausknopfes erzeugt sein könnte. Falls nicht, dann begibt sich die Routine auf die Lauer am Briefkasten ("WaitPort"). Dabei wird im Unterschied zur direkten Abfrage des Mausknopfes (wie in Teil 1 des Kurses) praktisch keine Rechenzeit "verbraten", obwohl es in der Wüste heiß ist. Nach Empfang einer Nachricht werden Teile extrahiert und der Erhalt bestätigt (ReplyMsg). Anschließend wird geprüft, ob die Nachricht über das Drücken des linken Mausknopfes ("cmp.l #MouseButtons") berichtet. Falls nein, wird die Routine erneut durchlaufen, sonst verlassen. Ganz schön aufwendig, aber multitasking-gerecht. Später werden wir in die Routine die Abfrage von Menüs integrieren.

Schwarz vor den Augen aber "reentrant"

Rein äußerlich hat unser Programm noch immer nichts gewonnen. Wir haben wiederum nur Ballast abgeworfen, z.B. die Routine "WarteTaste" und sehen außerdem noch schwarz. Nicht weil ein Sandsturm aufkommt oder der Strom ausfiel, sondern weil wir den Screen schwarz einfärben, wie Sie der Routine "ScreenAnlegen" entnehmen können. Dazu schnappen wir uns den "ViewPort" des Screen und trocknen ihn wüstengerecht aus, indem wir dem Hintergrund sämtliche Farbanteile entziehen. Was, Sie haben noch nicht assembliert und gebunden? Bei Druck auf den linken Mausknopf lichtet sich das Dunkel.

Hilfe, unsere PaintBox ist resident und reentrant zugleich. Was soll das, wir haben doch nur einen Bildschirm? Naja, ein Assemblerkurs sollte auch etwas Neues bringen, und davon können Sie sich nun überzeugen. Laden Sie Ihre PaintBox mit der Anweisung "resident bin/PaintBox pure" (oder wie immer Ihr Programm heißt) in den Arbeitsspeicher. Starten Sie es wiederholt! Sie werden feststellen, daß es ohne Zugriff auf den externen Speicher aufgestartet wird. Die "reentrancy" können Sie begutachten, indem Sie das Programm mehrfach mit "run" aufstarten. Es legen sich schwarze Screens übereinander, die Sie vertikal verschieben und mit Mausklicks schließen können. Lieben Sie es nicht so eintönig, dann könnten Sie in "Main" vor Aufruf von "ProgramControl" das "ZickZack" auf den Bildschirm schleudern lassen ("brs ZickZack"). Jeder mit "run" angelegte Prozeß nutzt denselben resident-geladenen Code. Das Multitasking und unsere Programmiertechnik machen es möglich. Zufrieden?

Haben Sie bemerkt, daß wir mit den Registern nun sehr viel sorgsamer umgehen, als in der ersten Folge? Bei jedem Eintritt in eine Routine werden die Register gerettet, auf die global schreibend nicht zugegriffen werden darf. Wir übertreiben dabei sogar ein wenig.

Tablett schnitzen

Listing 7 stellt das Tablett dar, auf dem unsere Anwendungen serviert werden können, z.B. "ZickZack". Nicht auszudenken, zickzack durch die Wüste! "MenuStuff.asm", kurz "M2", stellt dem Hauptprogramm zwei Routinen zur Verfügung: "MenuAnlegen" und "MenuLoeschen". Dazu setzen wir am Schluß von "M2" den Pseudocode "XDEF" ein. Wir hätten, wie bisher, auch "PUBLIC" nehmen können.

Da wir gerade den Schlußteil von "M2" betrachten, richten wir Ihr Augenmerk auf den üblichen Teil zur Deklaration von Menüstrukturen, in denen gewöhnlich "dc" regiert. Sie finden dort nur statische Daten, Menübezeichnungen und Tastencodes. Wir können hier kein Menü definieren, ohne unsere gerade erworbenen Fähigkeiten, reentrant und resident zu sein, wieder zu verlieren. Erkennen Sie jetzt, daß unser Wüstenauto mit einem Rad in einer tiefen Spur läuft? Menüstrukturen sind nicht statisch. Intuition schreibt hinein und auch wir setzen manchmal "Häkchen", um Unterpunkte zu kennzeichnen.

Es ist offensichtlich, daß wir für die Menüs Speicher anfordern und später wieder freigeben müssen. Die dafür benötigten Strukturen finden sich in "Intuition.i". Eine eigene Speicherverwaltung ist nicht erforderlich. Wir setzen die Routinen "AllocRemember" und "FreeRemember" aus Intuition ein. Für den ersten Aufruf von "AllocRemember" muß der Zugriff zur Speicherliste (RemKey) mit null initia-

Register	Farbe	Register	Farbe
0	blau	8	braun
1	weiß	9	rot
2	schwarz	10	gelb-grün
3	orange	11	gelb
4	mittl. blau	12	hellblau
5	violett	13	lila
6	türkis	14	hellgrün
7	weiß	15	grau

Tabelle 2

lisiert sein. Beim Anlegen der globalen Variablen haben wir das bereits erledigt.
Beim Anlegen des Menüs gehen wir wir folgt vor:

1. Speicher zur Aufnahme einer Struktur (Menu, MenuItem bzw IntuiText) anfordern
2. Struktur erstmalig initialisieren und davon ausgehen, daß sie nicht mit weiteren verkettet ist ("ErstesMenuInit", "ErstesMenuItemInit" bzw. "ErsterTextInit")
3. Weitere Ausprägungen der benötigten Strukturen anlegen durch Speicheranforderung, Kopieren des Inhaltes einer bereits bestehenden Struktur, Anpassen des Strukturinhaltes und Verketten des Elementes mit der gerade vorher angelegten Struktur ("WeiteresMenuInit", "WeiteresItemInit" bzw. "WeitererTextInit").

Damit ist Ihnen ein mächtiges Werkzeug an die Hand gegeben, ausgedehnte Menüstrukturen anzulegen. Um die Verbindung der Teile müssen Sie sich keine Sorgen machen und die Speicherverwaltung erfolgt mit minimalem Aufwand. Auf "SubItems" sind wir nicht eingegangen. Die Programmierung kann nach dem Muster "MenuItem" erfolgen.
Zu erwähnen sind noch die beiden Routinen "MenuAnheften" und "MenuLoeschen". Sie verbinden das erstellte Menü mit unserem Window bzw. hängen es wieder ab. Lassen wir "M2" nun nicht mehr länger im Fahrtwind trocknen. Werfen wir es dem Assembler zur Begutachtung vor. Es müßte alles glattgehen. Um das Menü zu betrachten, ist es erforderlich im Hauptteil von "P2" die Aufrufe "bsr MenuAnlegen", "bsr MenuLoeschen" und "bsr SpeicherListe" sowie "PUBLIC MenuAnlegen" und "PUBLIC MenuLoeschen" zu entsperren, indem wir die "Kommentarsterne" entfernen. Weiterhin müssen wir in "P2" noch die Routine "LoescheSpeicherListe" (Listing 8) einfügen, am besten vor "ProgramControl". Nun können wir "M2" mit unseren bisherigen Programmteilen binden und "P2" aufrufen. Das Tablett erscheint, aber es ist leer, und unser Programm reagiert noch nicht darauf. Offensichtlich ist noch etwas mehr zu tun, obwohl die Oase, unser Ziel, schon am Horizont auftaucht.

```
a. In  P2 ist "ZickZack" unbekannt. In "P2" und in "G2" ist
   deshalb  "PUBLIC  ZickZack"  aufzunehmen, um den "Export"
   und  den "Import" zu regeln. b. In "Intuition.i" sind die
   neuen  Makros  noch  unbekannt.  Die  Anweisung  "include
   Types.i" ist aufzunehmen.
c. GraphBase(globals) bzw. IntuiBase(globals)
d. Im   Prinzip   gelten   "GraphBase",   "IntuiBase"  und
   "ScreenTitle"  gleichfalls  als Variablen, die jedoch nur
   gelesen  werden,  somit  eigentlich  Konstanten sind. Die
   Objekte  bleiben  somit unverändert. Ihre Implementierung
   kann so besonders effizient erfolgen.
e. Der "interne Zeiger" ist in "Types.i" definiert. Er trägt
   den Namen "struktZeig".
```

Tabelle 3

Es wird serviert

Kurz vor dem Stall laufen die Kamele schneller. Listing 9 enthält alles, was wir benötigen, um das "Projekt-Menü" auszuwerten. Fügen Sie bitte den Code in "P2" vor der Routine "ProgramControl" ein. Ja, Sie haben richtig gesehen, mit dem ersten MenuItem können Sie eine Zeichenroutine aufrufen, nachdem vorher der Bildschirm gelöscht wurde. Der zweite Menüpunkt zaubert "ZickZack" auf den Schirm. Aber warum klappt das alles noch nicht? Es fehlt noch die Verbindung zwischen Intuition und unseren Routinen zur Auswertung der Menüs. Kein Problem, wir vertrösten Sie an der spannendsten Stelle nicht auf die nächste Folge. Listing 10 enthält fast alles, was wir dazu benötigen. Fügen Sie den Code in die Routine "ProgramControl" ein und zwar zwischen dem Statement "CallExec ReplyMsg" und dem Label "MouseBut". Letztlich müssen wir noch einmal die Routine "WindowAnlegen" anpassen, indem wir "IDCMPFL = MenuePick" setzen.

Fertig! Wir haben unser Etappenziel erreicht und den Kursfahrplan mehr als eingehalten. Sie können sogar die Wüste begrünen, indem Sie das MenuItem "Laden" anwählen und anschließend den linken Mausknopf drücken. Auf die Menüauswertung gehen wir in der nächsten Folge ausführlich ein. Als Aufgabe geben wir Ihnen bis dahin mit, in die Paintbox zwei weitere Menüs einzubauen und die Menüs eins bis drei mit jeweils vier Untermenüs zu versehen. So in etwa wird Ihr Bedarf aussehen, um weitere Routinen in die Programmsteuerung "einzuhängen".

Als weitere Aufgabe schlagen wir Ihnen vor, sich mit folgenden Fragen zu beschäftigen, bei deren Lösung jedes Assemblerhandbuch zur CPU 68000 (z.B. [3] oder [4]) wertvolle Hilfe leisten kann:

1. Wie erfolgt die Parameterübergabe bei Aufruf der Routinen "Menu0Auswerten" und "Menu1Auswerten"?

2. Betrachten Sie "Menu1Auswerten". Zeichnen Sie die Belegung des Stack auf. Vollziehen Sie jede Anweisung der Routine auf Ihrer Skizze nach. Wohin erfolgt der Sprung mit "jmp (a0)"? Wie ist die Adresse auf den Stack gekommen?

3. Betrachten Sie die Routine "LeftButtonPressed". Was bewirkt "link a2, #lb__size"? Warum haben die Offsets der lokalen Variablen negative Werte? Welche Rolle spielt das Register "a2" innerhalb der Routine. Was bewirkt am Ende der Routine die Anweisung "unlk a2"?

Literaturhinweise:

[1] Bleek, Jennrich, Schulz, "Amiga Intern Band 2"

[2] "Dr. Dobbs's Toolbook of 68000 Programming"

[3] Kief, "Assemblerprogrammierung mit dem M68000"

[4] Ostermann, "Maschinen-und Assemblersprache des M68000"

(mm)

Listings

```
* Makros, um Datenstrukturen anzulegen

* Die Makros dienen dazu, Offsets fur einzelne Elemente
* von Datenstrukturen festzulegen und die Lange der
* jeweiligen Struktur zu ermitten, um den dafur benotigt
en
* Speicherplatz zu reservieren. Uber die Offsets ist
* es moglich, die Elemente schnell und anschaulich zu
* adressieren (Adressregister indirekt mit Offset).

sizeByte = 1      ; Langen von Datentypen der CPU
sizeWord = 2
sizeLong = 4

* Einleitung einer Strukturdeklaration

struktur macro
struktZeig set 0 ; setzt internen Zeiger auf null, denn
         endm    ; das erste Element hat keinen Versatz
                 ; Der interne Zeiger enthalt stets die
                 ; aktuelle Lange der Struktur.



* Einfugen eines Strukturelementes, Lange ein Byte

byte     macro
\1           =   struktZeig
                     ; der ubergebene Name symbolisiert eine
n
                     ; Versatz und nimmt den aktuellen Wert
                     ; des internen Zeigers an.
struktZeig set struktZeig+sizeByte
                     ; internen Zeiger um Lange des Datentyp
s
                     ; erhohen (hier 1)
         endm


* Einfugen eines Strukturelementes, Lange ein Wort

word     macro
\1           =   struktZeig
struktZeig set struktZeig+sizeWord ; zusammenhangende
                     ; Schreibweise fur Funktion des Assembler
s
```

```
         endm   ; wichtig


* Einfugen eines Langwortes in eine Struktur

long      macro
\1        =    struktZeig
struktZeig set struktZeig+sizeLong
          endm

aptr      macro           ; Adressen sind auch Langworte
            long \1
          endm


* Einfugen eines beliebig langen Speicherblockes in eine
* Struktur

strukt  macro ; zur Aufnahme beliebig langer Strukturen
                ; in die Strukturdeklaration
\1      =    struktZeig
struktZeig set struktZeig+\2 ; Para 2 enthalt Blocklange
          endm


* Lange einer Struktur zum Abschlu3 der Deklaration er-
* mitteln

struktLaenge macro
\1          =    struktZeig
               endm

Listing 1: Auch in Assembler la3t sich strukturiert
             programmieren


```

Listing 1

```


           include "incl/Types.i"

* aus Intuition:

* Offsets der Sprungvektoren

AllocRemember   = -396 ; (rememberkey,size,flags)
*                      ; (a0, d0, d1)
ClearMenuStrip  = -054 ; (window) (a0)
CloseScreen     = -066 ; (screen) (a0)
CloseWindow     = -072 ; (window) (a0)
DisplayBeep     = -096 ; (screen) (a0)
FreeRemember    = -408 ; (rememberkey,reallyforget)
*                      ; (a0, d0)
ModifyIDCMP     = -150 ; (window,flags) (a0,d1)
OpenWindow      = -204 ; (windowData) (a0)
OpenScreen      = -198 ; (screenData) (a0)
SetMenuStrip    = -264 ; (window,menu) (a0,a1)

* Window-Flags

BackDrop      =    $100    WindowActive  =     $2000
ReportMouse   =    $200    WindowRefresh = $1000000
Activate      =   $1000

*IDCMP-Flags

MouseButtons  =      $8    ReqClear      =    $1000
MouseMove     =     $10    MenuVerify    =    $2000
MenuePick     =    $100    DeltaMove     =  $100000
ReqVerify     =    $800

* Screen Modes

sprites = $4000            hires    = $8000 ; 640

* Screen Types

WBenchScreen = $1          CustomScreen = $F

* MenuFlags

MenuEnabled = $1

* Menu Item Flags

checkIt     =   $1         menutogle   =  $8
itemText    =   $2         itemEnabled = $10
commSeq     =   $4

highFlags   = $C0          highComp    = $40
highImage   =    0         highBox     = $80

* andere Konstanten

SelectDown = $068

* Makros

CallIntui macro
   move.l a6,-(sp)
   move.l IntuiBase(globals),a6
   jsr \1(a6)
   move.l (sp)+,a6
   endm

* Datentypen aus Intuition

* IntuiMessage

 struktur
  strukt im_ExecMessage,20      word     im_MouseY
  long    im_Class              long     im_Seconds
  word    im_Code               long     im_Micros
  word    im_Qualifier          aptr     im_IDCMPWindow
  aptr    im_IAddress           aptr     im_SpecialLink
  word    im_MouseXm           struktLaenge im_size

* IntuiText

 struktur
  byte    it_frontPen           word    it_topEdge
  byte    it_backPen            aptr    it_iTextFont
  byte    it_drawMode           aptr    it_iText
  byte    it_allign             aptr    it_nextText
  word    it_leftEdge          struktLaenge it_size


* MenuItem

 struktur
  aptr    mi_nextItem            aptr    mi_itemFill
  word    mi_leftEdge            aptr    mi_selectFill
  word    mi_topEdge             byte    mi_command
  word    mi_width               byte    mi_allign
  word    mi_height              aptr    mi_subItem
  word    mi_flags               word    mi_nextSelect
  long    mi_mutualExclude     struktLaenge mi_size

* Menu

  struktur
     aptr me_nextMenu               word me_flags
     word me_leftEdge               aptr me_menueName
     word me_topEdge                aptr me_firstItem
     word me_width                  long me_jazzXY
     word me_height                 long me_beatXY
                                   struktLaenge me_size

* NewScreen     deklariert die NewScreen-Struktur zum
*               Anlegen eines neuen Screen
  struktur
     word       ns_leftEdge    ; Offset  0
     word       ns_topEdge     ; Offset  2
     word       ns_width       ; Offset  4
     word       ns_height
     word       ns_depth
     byte       ns_detailPen
     byte       ns_blockPen
     word       ns_viewModes   ; Offset 12
     word       ns_screenType
     aptr       ns_font        ; Offset 16
     aptr       ns_title
     aptr       ns_gadgets
     aptr       ns_bitmap      ; Offset 28
  struktLaenge ns_size         ; Lange  32


* NewWindow     deklariert die NewWindow-Struktur zum
*               Anlegen eines neuen Fensters

  struktur
     word       nw_leftEdge      aptr       nw_title
     word       nw_topEdge       aptr       nw_screen
     word       nw_width         aptr       nw_bitMap
     word       nw_height        word       nw_minWidth
     byte       nw_detailPen     word       nw_minHeight
     byte       nw_blockPen      word       nw_maxWidth
     long       nw_idcmpFlags    word       nw_maxHeight
     long       nw_flags         word       nw_type
     aptr       nw_firstGadget   struktLaenge nw_size
     aptr       nw_checkMark


* Screen-Struktur konnte hier definiert sein

sc_viewPort = $2C ; gehort zu Screen


* Window

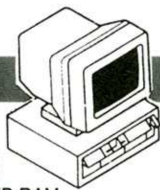
 struktur
  aptr   wd_NextWindow         aptr   wd_Pointer
  word   wd_LeftEdge           byte   wd_PtrHeight
  word   wd_TopEdge            byte   wd_PtrWidth
  word   wd_Width              byte   wd_XOset
  word   wd_Height             byte   wd_YOffset
  word   wd_MouseY             long   wd_IDCMPFlags
  word   wd_MouseX             aptr   wd_UserPort
  word   wd_MinWidth           aptr   wd_WindowPort
  word   wd_MinHeight          aptr   wd_MessageKey
  word   wd_MaxWidth           byte   wd_DetailPen
  word   wd_MaxHeight          byte   wd_BlockPen
  long   wd_Flags              aptr   wd_CheckMark
  aptr   wd_MenuStrip          aptr   wd_ScreenTitle
  aptr   wd_Title              word   wd_GZZMouseX
  aptr   wd_FirstRequest       word   wd_GZZMouseY
  aptr   wd_DMRequest          word   wd_GZZWidth
  word   wd_ReqCount           word   wd_GZZHeight
```



CPS Computertechnik GmbH
Hamburger Str. 283 · 3300 Braunschweig
Fax (05 31) 33 06 61 · BTX *20088 1490#

Ladenöffnungszeiten: Mo-Fr 9-18 Uhr
Sa 9-13 Uhr
Studentenrabatte geg. Nachweis a. Anfrage!

Tel. (05 31) 33 06 63



CPS AT

CPS AT I (siehe Test DOS 1/89)

80286-12 CPU 8/12 MHz, 1 MB RAM, bis 4 MB on board. 0-Wait-State, Sockel für Co-Prozessor, EMS, Phoenix Bios, 64 K ROM mit Setup-Funktion, Uhr u. Kalender, 1 ser./2 par./Game-Port, Herc. komp. Grafikkarte, 20 MB Festplatte 38 ms Zugriffszeit, 1 Lw 5,25" 1,2 MB und 1 Lw 3,5" 720 KB/1,44 MB Cherry Tastatur mit sep. Cursorblock, 14" Flat Screen Monitor s/w oder amber, MS-DOS 4.01 dt. und GW-Basic **3360,-**

CPS AT II

80286-16 MHz CPU. 12/16 MHz. 1 MB RAM, bis 4 MB on board, 0-Wait-State, EMS, Sockel für Co-Prozessor 80287, Bios 64 K ROM mit Setup-Funktion, Uhr u. Kalender, 1 ser./ 2 par./ Game-Port, VGA-Grafikkarte, 20 MB Festplatte 38 ms, 1 Lw 5,25" 1,2 MB u. 1 Lw 3,5" 720 KB/1,44 MB Cherry Tastatur mit sep. Cursorblock, 14" VGA Color Monitor MS-DOS 4.01 dt. und GW-Basic **4360,-**

CPS 386 SX

80386 SX CPU 16 MHz, 0-Wait-State, 2 MB RAM, bis 4 MB on board, Sockel für 80387, VGA-Grafikkarte. 40 MB Festplatte 28 ms 1 ser./1 par./ Gameport, 1 Lw 5,25" 1,2 MB und 1 Lw 3,5" 720 KB/1,44 MB Cherry Tastatur mit sep. Cursorblock, 14" VGA Color Monitor MS-DOS 4.01 dt. und GW-Basic **4955,-**

CPS AT III / Cache

80386-25 MHz CPU. 20/25 MHz. 0-Wait 2 MB RAM, erw. bis 16 MB on board, 32 KB Cache. Memory Expans. Card Sockel für Co-CPU 80387, EMS-fähig, 2 par./2 ser. Schnittst./ Gameport, 1 Lw 5,25" 1,2 MB und 1 Lw 3,5" 1,44 MB, VGA Grafik EIZO MD-B 10, SCSI Combi Controller 157 MB SCSI Festpl. 14 ms 14" EIZO 9070s Monitor MS-DOS 4.01 und GW-Basic UNIX V/386 OS **15.500,-**

Technische Änderungen vorbehalten. Gerätekonfigurationen nach Ihren Wünschen veränderbar.

CPS-Artikel erhalten Sie auch bei:

CPS Computertechnik GmbH
Braunhirschstraße 29, 3100 Celle
Tel. 0 51 41 / 3 20 04, Fax 0 51 41 / 38 14 22

CPS Computertechnik GmbH
Großbeerenstraße 5 · D-1000 Berlin 42
Telefon 0 30 / 705 30 55

COMMODORE

PC 10 III, 1 Laufwerk 360 K, Festplatte 20 MB, EGA Grafikkarte, EGA Colormonitor 14" Quadram, MS-DOS 3.3 / GW-Basic, Drucker Panasonic KX-P 1081 .... **2999,-**

COMMODORE

AMIGA 500	**795,-**
AMIGA 500 + 1084	1399,-
AMIGA 2000	1985,-
AMIGA 2000 + 20 MB Harddisk Autoboot inkl. Contr. 2090 A	2990,-
AMIGA 2000 + Monitor 1084	2480,-
68020 Prozessorkarte	Preis a. Anfr.
68030 Prozessorkarte	Preis a. Anfr.

ERWEITERUNGEN

20 MB Harddisk A 2000 inkl. 2090 A Contr.	1220,-
2 MB Erw. int. (A 2000) orig. CBM	1180,-
PC/XT Karte inkl. 5,25" LW + MS-DOS + GW-Basic	899,-
PC/AT Karte inkl. 5,25" Lw + MS-DOS + GW-Basic	2400,-
512 KB Erw. A 500	330,-

AMIGA-ZUBEHÖR

LW ext. 3,5" ohne Display	290,-
LW ext. 5,25" ohne Display	419,-
LW intern 3,5" inkl. Einbausatz	220,-
Commodore 1084	630,-
Philips RGB Color CM 8833	630,-
EGA Standard	815,-



DRUCKER

OKI Microline 390 Centr.	Sonderpreis auf Anfr.
OKI Microline 393 Centr.	Sonderpreis auf Anfr.
OKI Microline 320	Sonderpreis auf Anfr.
Star LC 10 Centr.	499,-
Epson LX 400	499,-
NEC P 6 Plus Centr.	1499,-
NEC P 7 Plus Centr.	1985,-
Centronics Druckerkabel	18,80

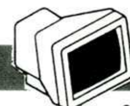
Weitere Drucker auf Anfrage

Wir liefern **nur** mit dt. Handbuch, Seriennummer und Herstellergarantie!!! Drucker-Grauimporte mit engl. Handbuch, ohne Seriennummer, ohne Herstellergarantie sind bei uns ausgeschlossen.

Nutzen Sie unseren Bequem-Kauf-Kredit!

CPS

weil Preis und Leistung stimmen!



MONITORE

EGA Standard	815,--
EIZO 9060s	1855,-
NEC MULTISYNC II A	1550,-
TTL 14" Flat Screen mit Fuß sw/amber	235,-
TTL 12" Flat Screen mit Fuß sw/amber	180,-
Mitsubishi 1481	1398,-

weitere Monitore auf Anfrage



DISKETTEN

NN 2DD 3,5" 10 Stück	15,00
NN 2DD 5,25" 10 Stück	6,90
Magix MF 2DD 3,5"	24,50
Select MF 2DD 3,5"	33,50
Select MD 2DD 5,25"	19,00
Fuji MD 2DD 5,25"	27,00



FARBBÄNDER

STAR NL/NG/ND/NR-10, Stück	8,60
EPSON LX-800/LQ-500, Stück	7,50
PANASONIC KX-P, Stück	9,25
OKI ML 320	7,20
OKI ML 390	10,40
NEC 2200	10,25
NEC P6+/P7+	10,90
Star LC 10	7,00

SOFTWARE

ALDUS Pagemaker 3.0	2150,-
GEM Artline 1.0	1390,-
GEM Desktop 1.1	1200,-
MS-DOS Lernprogramm 2.0	168,-
MS-Windows 286 2.1	480,-
MS-Windows 386 2.1	750,-

weitere Software auf Anfrage

Sämtliche Angebote freibleibend, Zwischenverkauf vorbehalten. Wir liefern an Nicht-Kaufleute nur per UPS-Nachnahme. Ins Ausland nur per Vorkasse. Fordern Sie unseren Gesamtkatalog gegen 3,- DM in Briefmarken. Es gelten unsere allgemeinen Geschäftsbedingungen.

Versand nur über Braunschweig!

- 24 Monate Garantie auf CPS-Rechner
- 10 Tage Rückgaberecht auf Hardware
- Eigener Reparatur-Service

Die von uns angegebenen Taktfrequenzen (MHz) beziehen sich **nicht** auf den Speedtest nach Landmark, sondern sind **tatsächliche** Taktfrequenzen!

```
    aptr   wd_WScreen              aptr   wd_ExtData
    aptr   wd_RPort                aptr   wd_UserData
    byte   wd_BorderLeft           aptr   wd_WLayer
    byte   wd_BorderTop            aptr   wd_IFont
    byte   wd_BorderRight         struktLaenge wd_size
    byte   wd_BorderBottom
    aptr   wd_BorderRPort
    aptr   wd_FirstGadget
    aptr   wd_Parent
    aptr   wd_Descendant


Listing 2: Intuition.i, fur den Transport gefaltet

```

Listing 2

```
* Programm Paint Box 0.2

                    include "incl/Types.i"
                    include "incl/Exec.i"
                    include "incl/Intuition.i"
                    include "incl/Graphics.i"

* aus anderen Objektdateien importieren:
* aus Grafik0.o:

  PUBLIC ZickZack

* aus MenuStuff

*  PUBLIC MenuAnlegen
*  PUBLIC MenuLoeschen

* Globale Variablen - Offsets innerhalb eines
*                     Speicherblockes

 struktur           ; Details siehe Types.i
  aptr GraphBase  ; Basisadr. Grahpics-Lib
  aptr IntuiBase  ; Basisadr. Intui-Lib
  aptr Menu       ; --> unser Menu
  aptr RemKey     ; Zugriff auf Memory-Liste
  aptr Screen     ; Adresse des Screen
  aptr RastPort   ; Adresse des Window.Rastport
 struktLaenge globals_size

* davon allen Modulen zuganglich machen:

  PUBLIC GraphBase
  PUBLIC IntuiBase
  PUBLIC Menu
  PUBLIC RemKey

* Registergebrauch

* aufgefuhrte Register sollen Inhalte wahrend der Laufze
it
* beibehalten. Bei zwischenzeitlichem Bedarf sind sie zu
* sichern und zu restaurieren

saved_Regs reg  a2-a6/d2-d7 ; "nicht-scratch" Register
globals      equr a5          ; --> Struktur Globale Vars
our_Win      equr a4          ; --> unser Window
our_Return equr d7            ; unser ReturnCode


* Unsere Return-Codes mit Hinweis auf Abbruchursache

kein_Speicher = -1
lib_Problem   = -2
kein_Screen   = -3
kein_Window   = -4

  XDEF kein_Speicher

  bra Main        ; Sprung uber Unterprogramme zum
                  ; Hauptteil

* Unterprogramme

LibsOeffnen
*----------
    OpenLib GraphLib,GraphBase(globals) ; Graph Lib
    beq.s \1                            ; erfolglos
    OpenLib IntuiLib,IntuiBase(globals) ; Intui Lib
\1 rts

LibsSchliess
*-----------
    CloseLib IntuiBase(globals) ; Intui Lib
    CloseLib GraphBase(globals) ; Graph Lib
    rts

ScreenAnlegen
*------------
NScrPtr equr a3 ; zeigt auf NewScreen-Struktur

    movem.l a3/a5-a6,-(sp)
    AllocMemory #ns_size,#Mem_Clear ; Screen-Beschreib
    tst.l d0                        ; Speicher da ?
    beq.s   SA                      ; nein
    move.l  d0,NScrPtr              ; --> Scr-Beschreib
    move.w  #640,ns_width(NScrPtr)  ; Breite
    move.w  #256,ns_height(NScrPtr) ; Hohe
    move.w  #4,ns_depth(NScrPtr)    ; 4 Bitplanes
    move.b  #weiss,ns_blockPen(NScrPtr)
    move.w  #hires,ns_viewModes(NScrPtr)
    move.w  #CustomScreen,ns_screenType(NScrPtr)
    lea.l   ScreenTitle(pc),a1      ; Name
    move.l  a1,ns_title(NScrPtr)

    movea.l NScrPtr,a0              ; Para-Ubergabe
    CallIntui OpenScreen            ; Screen oeffnen
    move.l  d0,Screen(globals)      ; global merken
*   Hintergrund schwarz einfarben

    move.l  d0,a0                   ; Screen Adr
    lea.l   sc_viewPort(a0),a0      ; Adr Viewport holen
    clr.w   d0                      ; Hintergrund
    clr.b   d1                      ; Rotanteil
    clr.b   d2                      ; Grunanteil
    clr.b   d3                      ; Blauanteil
    CallGraph SetRGB4               ; schwarzer Hintergrund

    FreeMemory NScrPtr,#ns_size ; nicht mehr gebraucht
SA movem.l (sp)+,a3/a5-a6
    rts

ScreenSchliess
*-------------
    movem.l a5-a6,-(sp)
    tst.l  Screen(globals)          ; Screen-Adr # null
    beq.s  \S                           ; null =  nichts tu
n
    move.l Screen(globals),a0       ; Parameterubergabe
    CallIntui CloseScreen           ; Screen schlie3en
\S movem.l (sp)+,a5-a6
    rts


WindowAnlegen
*------------
NWinPtr equr a3 ; zeigt auf NewWindow-Struktur
ourFlgs = BackDrop¦Activate¦ReportMouse
IDCMPFL = MouseButtons ; ¦MenuePick

    movem.l a3/a5-a6,-(sp)
    AllocMemory #nw_size,#Mem_Clear ; Window-Beschreib
    tst.l   d0                      ; Speicher da ?
    beq.s   WA                      ; nein
    move.l  d0,NWinPtr                ; --> Window-Def
    move.w  #10,nw_topEdge(NWinPtr)   ; Oberkante
    move.w  #640,nw_width(NWinPtr)    ; Breite
    move.w  #246,nw_height(NWinPtr)   ; Hohe
    move.b  #weiss,nw_detailPen(NWinPtr)
    move.l  #IDCMPFL,nw_idcmpFlags(NWinPtr)
    move.l  #ourFlgs,nw_flags(NWinPtr)
    move.l  Screen(globals),nw_screen(NWinPtr); unser Scr
    move.w  #CustomScreen,nw_type(NWinPtr)

    movea.l NWinPtr,a0              ; Para-Ubergabe
    CallIntui OpenWindow            ; Window oeffnen
    move.l  d0,our_Win                ; Zeiger auf Win
dow
    move.l  wd_RPort(our_Win),RastPort(globals)
    FreeMemory NWinPtr,#nw_size     ; hat ausgedient
WA movem.l (sp)+,a3/a5-a6
    rts

WindowSchliess
*-------------
    movem.l a5/a6,-(sp)
    move.l  our_Win,d0
    beq.s   \W                          ; null =  nichts t
un
    move.l  our_Win,a0                  ; Parameterubergab
e
    CallIntui CloseWindow           ; Schliesse Window
\W movem.l (sp)+,a5/a6
    rts


ProgramControl ; PC
*-------------
ourPort  equr a3
MsgClass equr d5 ; Register hat zwei Namen !
ItemNr   equr d5
MsgCode  equr d6 ; Register mit zwei Namen !
MenuNr   equr d6

    movem.l d5-d6/a3,-(sp)
    move.l  wd_UserPort(our_Win),ourPort
control0
    movea.l  ourPort,a0         ; ParaUbergabe
    CallExec GetMsg             ; Nachricht holen
    tst.l    d0                 ; Nachricht da ?
    bne.s    \controll          ; ja
    move.l   ourPort,a0         ; Paraubergabe
    CallExec WaitPort           ; auf Nachrichht warten
    bra.s    control0           ; Nachricht erhalten
\controll
    move.l   d0,a1              ; Antwort vorbereiten
    move.l   im_Class(a1),MsgClass ; Info extrahieren
    move.w   im_Code(a1),MsgCode
    CallExec ReplyMsg           ; Erhalt bestatigen

MouseBut
    cmp.l    #MouseButtons,MsgClass ; Nachricht auswerten
*   hier Aktion (bei gedruckten Buttons) einsetzen
```

```
        bra.s     control2          ; spater entfernen !!!!!!!!
        bra       control0          ; Nachricht abholen
control2
        movem.l (sp)+,d5-d6/a3
        rts
* Hauptprogramm
Main
        movem.l saved_Regs,-(sp) ; non-scratch sichern
* Globale Vars anlegen, mit null initialisieren
        moveq #kein_Speicher,our_Return  ; eigener Code
        AllocMemory #globals_size,#Mem_Clear
        tst.l d0                    ; haben wir Speicher
        beq.s endeM                 ; nein
        move.l d0,globals           ; globals angelegt
* Libraries offnen
        move.l #lib_Problem,our_Return
        bsr LibsOeffnen    ; Libraries offnen
        beq.s endeL        ; Fehler aufgetreten ?
* Screen anlegen
        move.l #kein_Screen,our_Return
        bsr ScreenAnlegen
        tst.l Screen(globals) ; Scren angelegt ?
        beq.s endeS            ; nein
* Window anlegen
        move.l #kein_Window,our_Return
        bsr WindowAnlegen
        move.l our_Win,d0   ; Window angelegt ?
        beq.s endeW         ; nein
*       bsr MenuAnlegen
        clr.l our_Return    ; kein Fehler aufgetreten
        bsr ProgramControl
*       bsr MenuLoeschen
*       bsr LoescheSpeicherListe
endeW
        bsr WindowSchliess ; alles schliessen, umge-
endeS                            ; kehrt wie geoeffnet
        bsr ScreenSchliess
endeL
        bsr LibsSchliess
        FreeMemory globals,#globals_size
endeM
        move.l our_Return,d0      ; Returncode setzen
        movem.l (sp)+,saved_Regs ; restaurieren
        rts                                ; Programmende
* Statische Daten
                  even
GraphLib:         dc.b 'graphics.library',0
                  even  ; Ende Strings mit 0 markiert
IntuiLib:         dc.b 'intuition.library',0
                  even
ScreenTitle:      dc.b ' Paint Box 0.2 ',0
                  end
Listing 3: Paint Box 0.2, resident und reentrant
```

Listing 3

```
execBase       =     4
* Offsets der Sprungvektoren
AllocMem       = -198 ; (byteSize,requirements) (d0,d1)
CloseLibrary   = -414 ; (library) (a1)
CopyMem        = -624 ; (src,dest,size) (a0,a1,d0)
FreeMem        = -210 ; (memoryBlock,byteSize) (a1,d0)
GetMsg         = -372 ; (port) (a0)
OpenLibrary    = -552 ; (libName, version) (a1,d0)
ReplyMsg       = -378 ; (message) (a1)
WaitPort       = -384 ; (port) (a0)
* MemReqs bestimmen die Art des angeforderten Speichers
Mem_Chip            =       2
Mem_Fast            =       4
Mem_Clear           = $10000
Mem_Largest         = $20000
* Makros
AllocMemory macro  ; Speicherplatz anfordern
        move.l \1,d0 ; Register s. AllocMem
        move.l \2,d1
        CallExec AllocMem
    endm
FreeMemory macro          ; Speicherblock zuruckgeben
        move.l \1,d0      ; ist etwas zuruckzugeben ?
        beq    \@         ; nein
        move.l d0,a1      ; Registergebrauch s. FreeMem
        move.l \2,d0
        CallExec FreeMem
\@      nop
    Endm
Listing 4: Exec.i ist zu erganzen
```

Listing 4

```
* aus Graphics:
* Offsets der Sprungvektoren
Draw          =  -246 ; (rp,x,y) (a1,d0,d1)
DrawEllipse   =  -180 ; (rp,XM,YM,Xr,Yr) (a1,d0-d3)
Flood         =  -330 ; (rp,x,y) (a1,d0,d1)
Move          =  -240 ; (rp,x,y) (a1,d0,d1)
RectFill      =  -306 ; (rp,x1,y1,x2,y2) (a1,d0-d3)
SetAPen       =  -342 ; (rp, colour) (a1,d0)
SetBPen       =  -348 ; (rp, colour) (a1,d0)
SetDrMd       =  -354 ; (rp, drmode) (a1,d0)
SetRast       =  -234 ; (rp, colour) (a1,d0)
SetRGB4       =  -288 ; (viewPort,reg,r,g,b) (a0,d0-d3)
WritePixel    =  -324 ; (rp,X,Y) (a1,d0,d1)
* Farben (Standard Farbregister)
wb_blau = 0
weiss   = 1
schwarz = 2
orange  = 3
* Draw Modes
jam1        = 0
jam2        = 1
complement  = 2
inversvid   = 4
CallGraph macro
*---------
    move.l a6,-(sp)
    move.l GraphBase(globals),a6
    jsr    \1(a6)
    move.l (sp)+,a6
    endm
* Datentypen aus Graphics
* RastPort
 struktur
    aptr rp_layer          word rp_flags
    aptr rp_bitMap         word rp_linePtrn
    aptr rp_areaPtrn       word rp_x
    aptr rp_tmpRas         word rp_y
    aptr rp_areaInfo       strukt rp_minTerms,8
    aptr rp_gelsInfo       word rp_penWidth
    byte rp_mask           word rp_penHeight
    byte rp_fgPen          aptr rp_font
    byte rp_bgPen          byte rp_algoStyle
    byte rp_aOlPen         byte rp_txFlags
    byte rp_drawMode       word rp_txHeight
    byte rp_areaPtSz       word rp_txWidth
    byte rp_linPatCnt      word rp_txBaseLine
    byte rp_dummy          word rp_txSpacing
                           aptr rp_user
                           strukt rp_longreserved,8
                           strukt rp_wordreserved,14
                           strukt rp_reserved,8
                        struktLaenge rp_size
Listing 5: Graphics.i, gleichfalls gefaltet
```

Listing 5

```
    include "incl/Intuition.i"
    include "incl/Graphics.i"
    PUBLIC GraphBase
    PUBLIC ZickZack
our_Win equr a4
globals equr a5
```

```
DoRpFunct macro ; Fuhre eine Grafikfunktion auf
                ; einem Rastport (Rp) aus.
                ; Param: Funktion
   move.l wd_RPort,a1 ; Rastport-Adresse holen
   CallGraph \1 ; siehe Graphics.i
   endm


Listing 6: So sollte der Anfang von G2.asm aussehen


```

Listing 6

```
* Modul MenuStuff

  include "incl/Types.i"
  include "incl/Intuition.i"
  include "incl/Exec.i"
  include "incl/Graphics.i"

  PUBLIC IntuiBase      ; alle PUBLIC sind Importe
  PUBLIC RemKey         ; aus dem Hauptprog.
  PUBLIC Menu
  PUBLIC kein_Speicher

* Registergebrauch

* global benutzte, d.h. Importe aus dem Hauptprog.

our_Win    equr a4
globals    equr a5
our_Return equr d7

* lokal (bezogen auf MenuStuff), zur Verkettung der
  ; Menusruktur

menu  equr a3 ; zuletzt erzeugtes Menu
item  equr a2 ; zuletzt erzeugtes MenuIttem
oText equr d6 ; zuletzt erzeugte IntuiText-Struktur

MenuBenennen macro
*-----------
   lea.l  \1(pc),a1
   move.l a1,me_menueName(menu)
   endm

StrukturKopieren macro
*---------------
   movea.l \1,a0              ; Quelle
   move.l  \2,a1              ; Ziel
   move.l  \3,d0              ; Anz. Bytes
   CallExec CopyMem           ; kopiere
   endm

MemBlockAnfordern ; Blockgoesse in d0
*-----------------   Speicherliste in RemKey(globals)
   move.l #kein_Speicher,our_Return ; Fehlercode setzen
   move.l RemKey(globals),a0        ; Zugriff Mem Liste
   move.l #Mem_Clear,d1             ; Speichertyp
   CallIntui AllocRemember
   beq.s \M
   move.l a0,RemKey(globals)        ; listenzugriff
\M rts

ErstesMenuInit ; Adr des Menus in menu und Menu(globals)
*-------------
   move.l #me_size,d0    ; Laenge Menu-Struktur
   bsr MemBlockAnfordern
   beq.s \E              ; nein
   move.l d0,menu                ; erzeugtes Menu
   move.l d0,Menu(globals)       ; fur --> Menustreifen
   move.w #2,me_leftEdge(menu)   ; in Menus verschieden
   move.w #100,me_width(menu)    ; Standardbreite
   move.w #10,me_height(menu)    ; Standard
   move.w #MenuEnabled,me_flags(menu) ; einheitlich
   MenuBenennen Menu_0
\E rts

WeiteresMenuInit ; Adr des Menus in menu
*---------------
   move.l #me_size,d0          ; Lange MenuItem
   bsr MemBlockAnfordern
   beq.s \W                    ; kein Speicher
   move.l d0,-(sp)             ; Adr erzeugtews Menu
   StrukturKopieren menu,d0,#me_size
   move.l  (sp),me_nextMenu(menu) ; erzeugt. Menu anfuge
n
   move.l  (sp)+,menu          ; --> auf erzeugtes Menu
   add.w   #100,me_leftEdge(menu) ; einheitl. Breite
   clr.l   me_nextMenu(menu)   ; (zunachst) kein Anschlu3
   clr.l   me_firstItem(menu)  ; (zunachst) kein SubItem
   move.l  menu,d0             ; condition code setzen
\W rts

ErsterTextInit ; Zeiger auf Struktur in oText
*-------------
   move.l #it_size,d0
   bsr MemBlockAnfordern
   beq.s \T
   move.l d0,oText ; sichern f. Strukt.Kopie/TextEintrag

   move.l d0,a1
   move.b #schwarz,it_frontPen(a1)
   move.b #weiss,it_backPen(a1)
   move.b #complement,it_drawMode(a1)
   move.w #10,it_leftEdge(a1)
   move.w #1,it_topEdge(a1)
\T rts

ItemTextEintragen macro ; \1 Textadr. \2 Command
*----------------
   lea.l \1,a0              ; Textadresse holen
   move.l oText,a1
   move.l a0,it_iText(a1) ; und eintragen
   move.b #\2,mi_command(item)
   endm

ourItemFlags = itemText¦highComp¦itemEnabled¦commSeq

ErstesMenuItemInit ; Ergebnis in item
*-----------------
   bsr ErsterTextInit
   beq.s \J
   move.l #mi_size,d0
   bsr MemBlockAnfordern
   beq.s \J
   move.l d0,item ; erzeugtes Item merken
   move.w #120,mi_width(item)
   move.w #10,mi_height(item)
   move.w #ourItemFlags,mi_flags(item)
   move.l oText,mi_itemFill(item)
   ItemTextEintragen Item00,Cmnd00
   move.l item,me_firstItem(menu)
\J rts

WeitererTextInit ; Ergebnis in oText
*---------------
   move.l #it_size,d0
   bsr MemBlockAnfordern
   beq.s \T
   move.l d0,-(sp)
   StrukturKopieren oText,d0,#it_size
   move.l (sp)+,oText
\T rts

WeiteresItemInit ; Ergebnis in Item
*---------------
   move.l #mi_size,d0
   bsr MemBlockAnfordern
   beq.s \i
   move.l d0,-(sp) ; --> auf erzeugte Struktur
   StrukturKopieren item,d0,#mi_size
   move.l (sp),mi_nextItem(item) ; Item anfugen
   move.l (sp),item              ; --> erzeugte Strukt.
   add.w #12,mi_topEdge(item)
   move.l oText,mi_itemFill(item)  .
   clr.l mi_nextItem(item); zunachst kein Anschlu3
   move.l (sp)+,d0
\i rts

MenuAnlegen
*----------
   movem.l a2-a6,-(sp)

   bsr ErstesMenuInit     ; Menu_0 anlegen
   tst.l Menu(globals)
   beq \M
   bsr ErstesMenuItemInit ; Item00
   beq \M
   bsr WeitererTextInit   ; Item01
   beq \M
   bsr WeiteresItemInit
   ItemTextEintragen Item01,Cmnd01
   beq \M
   bsr WeitererTextInit   ; Item02
   beq \M
   bsr WeiteresItemInit
   ItemTextEintragen Item02,Cmnd02
   beq \M

   bsr WeiteresMenuInit   ; Menu_1 anlegen
   beq.s \M
   MenuBenennen Menu_1

   bsr MenuAnheften       ; an das Window
\M movem.l (sp)+,a2-a6
   rts

MenuAnheften
*-----------
   move.l our_Win,a0
   move.l Menu(globals),a1
   CallIntui SetMenuStrip
   rts

MenuLoeschen
*-----------
   move.l our_Win,a0
   CallIntui ClearMenuStrip
   rts
```

```
    XDEF MenuAnlegen  ; fur Aufrufe aus Hauptprog.
    XDEF MenuLoeschen

* Menu-Bezeichnungen

Menu_0  dc.b '  Projekt',0
Item00  dc.b ' Laden',0
Cmnd00  =      '1'
Item01  dc.b ' Sichern',0
Cmnd01  =      's'
Item02  dc.b ' Beenden',0
Cmnd02  =      'q'

Menu_1  dc.b '  Zeichnen',0
          end


Listing 7: Leistungsfahige Routinen zum Anlegen
           von Menustrukturen
```

Listing 7

```
LoescheSpeicherListe
*--------------------
   tst.l RemKey(globals)      ; etwas zu loschen ?
   beq.s \L                   ; nein
   move.l RemKey(globals),a0  ; Zugriff holen
   Move.b #$FF,d0             ; alles loschen
   CallIntui FreeRemember
\L  rts

Listing 8: Den Speicherplatz der Menustruktur auf
           einmal freigeben
```

Listing 8

```
LoescheHintergrund
*------------------
   move.l wd_RPort(our_Win),a1
   move.b rp_bgPen(a1),d0
   CallGraph SetRast
   rts


LeftButtonPressed ; Ergebnis in d0
*-----------------
* Lokale Variablen
lb_inMsg   = -4   ; hier nicht verwendet (Demo)
lb_pressed = -5   ; Knopf gedruckt ?
lb_size    = -6   ; 6 Bytes gebraucht (Wortgrenze)

    move.l d6,-(sp)           ; intern benotigt
    link   a2,#lb_size        ; Platz auf Stack reservieren
                              ; a2 auf Stack sichern
    clr.b  lb_pressed(a2)     ; zunachst ungedruckt annehme
n
    move.l wd_UserPort(our_Win),a0 ; fur Nachrichten
    move.l a0,-(sp)           ; gleich wieder gebraucht
    CallExec WaitPort         ; warten bis Nachr. da

    move.l (sp)+,a0           ; Userport holen
    CallExec GetMsg           ; Nachricht abholen
    tst.l d0                  ; gultige Nachricht ?
    beq \Lb                   ; nein

*    move.l d0,lb_inMsg(a2)   ; fur weitere Bearbeit.
    move.l d0,a1              ; Antwort vorbereiten
    move.l im_Class(a1),d6
    btst   #3,d6              ; MouseButton gedruckt
    beq \Re                   ; nein
    cmpi.w #SelectDown,im_Code(a1) ; link. Button ?
     seq d0                   ; FF falls ja
    move.b d0,lb_pressed(a2)  ; Ergebnis merken
\Re
     CallExec ReplyMsg        ; Nachricht beantworten
\Lb
    move.b lb_pressed(a2),d0  ; Ergebnis holen
    andi.l #$FF,d0            ; unteres Byte isolieren
    unlk a2                   ; lokalen frame zuruck
    move.l (sp)+,d6           ; restaurieren
     rts

idcmp_Flags = DeltaMove¦MouseMove¦MouseButtons

FreeDraw
*-------
* lokale Variable
fd_idcmp   = -4
fd_pressed = -5
fd_size    = -6

    link a2,#fd_size ; lokalen frame, a2 sichern

    movea.l RastPort(globals),a1 ; Zeichenfarbe setzen
    move.w  #14,d0               ; 1 - 15
    CallGraph SetAPen

    movea.l RastPort(globals),a1 ; Zeichenmodus best.
    move.b #jam1,d0              ; mit einer Farbe
    CallGraph SetDrMd

    move.l wd_IDCMPFlags(our_Win),fd_idcmp(a2) ; merken
    move.l wd_IDCMPFlags(our_Win),d0
    ori.l #idcmp_Flags,d0        ; Flags erganzen
    movea.l our_Win,a0
    CallIntui ModifyIDCMP        ; Flags andern

\encore                     ; warten bis 1.But. gedruckt
    bsr LeftButtonPressed
    tst.b d0
    bne.s \1
    bra.s \encore
\1                          ; Beginn Zeichenschleife
    bsr LeftButtonPressed   ; Abfragen lin. But. gedruckt
    tst.b d0
    beq.s \x                ; nein, also zeichnen
    bra.s \f                ; ja, somit fertig
\x
    move.w wd_MouseX(our_Win),d0 ; einen Punkt setzen
    move.w wd_MouseY(our_Win),d1
    movea.l RastPort(globals),a1
    CallGraph WritePixel
    bra.s \1                ; an den Anfang der Schleife
\f
    move.l fd_idcmp(a2),d0 ; IDCMP zuruckstellen
    movea.l our_Win,a0
    CallIntui ModifyIDCMP

    unlk a2                 ; Frame aufgeben
    rts

Menu0Auswerten ; Item auf Stack
*---------------
    move.w 4(sp),d0 ; ItemNr holen
    bne.s \1         ; 0 nicht, evtl. 1
    bsr LoescheHintergrund
    bsr FreeDraw
\1 cmp.w #1,d0      ; Item 1 ?
    bne.s \e         ; auch nicht
    bsr ZickZack
\e move.l (sp)+,a0 ; Rucksprungadr. holen
    addq.l #2,sp     ; Stack berichtigen
    jmp    (a0)      ; aufruf. Prog. fortsetz.

Menu1Auswerten ; Item auf Stack
*---------------
    move.w 4(sp),d0 ; ItemNr holen
    move.l (sp)+,a0
    addq.l #2,sp
    jmp    (a0)


Listing 9: Auswerten von Menus und Zeichnen
           mit der Maus
```

Listing 9

```
    cmp.l    #MenuePick,MsgClass ; ein Menu angewahlt ?
    bne.s    MouseBut            ; nein

    move.w   MsgCode,-(sp)       ; Copy fur Item-Auswertung
    and.w    #$1F,MenuNr         ; Menu Nr isolieren
    move.w   (sp)+,ItemNr        ; Code holen
    asr.w    #5,ItemNr           ; Menu Nr herausschieben
    and.w    #$3F,ItemNr         ; Item Nr isolieren

    tst.w    MenuNr              ; Menu0 angewahlt ?
    bne.s    \M1                 ; nein, evtl. 1
    cmpi.w   #2,ItemNr           ; quit ? (aus Menu0)
    beq      control2            ; ja
    move.w   ItemNr,-(sp)        ; Parameterubergabe
    bsr      Menu0Auswerten      ; nein, volle Auswertung
    bra      control0
\M1
    btst     #0,MenuNr           ; Menu1 angewahlt ?
    bne      control0            ; nein, Nachricht abholen
    move.w   ItemNr,-(sp)        ; Parameterubergabe
    bsr      Menu1Auswerten
    bra      control0            ; Nachricht abholen

Listing 10: Wesentliche Erganzung der Programm-
            steuerung, um Menus auszuwerten
```

Listing 10

Michael Winterberg

Soundeditor in Assembler

Grundlagen der Musikprogrammierung (Teil 2)

Sie erinnern sich? Beim letzten Mal haben wir gelernt, wie man auf dem Amiga einen Ton ausgibt und dessen Lautstärke verändert. Schließlich war es sogar schon möglich, kleinere (wenn auch einfache) Musikstücke zu erstellen. In diesem Kursteil wollen wir uns einer Routine widmen, die ein Musikstück per Interrupt abspielt.

Der geeignetste Interrupt für solch eine Routine ist der sogenannte Vertical-Blank-Interrupt. Er wird gestartet, wenn der Beginn der vertikalen Austastlücke erreicht wird, also sozusagen jedesmal, wenn der Rasterstrahl des Monitors auf die Zeile null (in der linken oberen Bildschirmecke) trifft. Der Interrupt wird so fünfzigmal pro Sekunde gestartet. Schnell genug, um Noten abzuspielen und die Lautstärke eines Tones zu verändern. Wie aber lassen wir anstatt des Interrupts unsere Routine starten?

Als erstes benötigen wir das Register, das den Zeiger auf der Original-Interrupt-Routine beinhaltet. Dieses Register befindet sich bei der Adresse $6C. Um die Musikabspielroutine starten zu lassen, brauchen wir nur den Zeiger auf unsere Routine “umbiegen“. Die alte Einsprungadresse des Interrupts sollten wir dabei jedoch unbedingt speichern, da, wenn unsere Abspielroutine beendet ist, auf jeden Fall die alte Interrupt-Routine angesprungen werden sollte, weil diese noch wichtige Daten verarbeiten muß. Bevor wir aber das Programm schreiben, wollen wir Ihnen noch einen kleinen Trick zeigen:

Wir wollen unsere Abspielroutine ja für jeden Speicherbereich lauffähig machen. Da ein DMA normalerweise aber nicht auf das Fast-RAM, sondern nur auf das Chip-RAM ($00000- $7FFFF) zugreifen kann, muß an dieser Stelle etwas in die Trickkiste gegriffen werden. Aus diesem Grunde reservieren wir einfach ein Teil des Chip-RAMs und kopieren uns die digitalen Daten in diesen Speicherbereich hinein. Die Routinen dazu erklären wir Ihnen gleich im Source-Code. Dieser ist diesmal übrigens ein wenig länger, dafür können Sie damit aber auch vierstimmige Kompositionen erstellen.

Herr, laß Chip-RAM regnen!

Fangen wir als erstes mit dem Hauptprogramm an, also mit der “Programm-Steuerung“. Bei manchen Amigas (z.B. beim 2000-B) kann man mittels Power-LED den Bit-Filter (zuständig für die Helligkeit der LED) ausschalten. Das hieße also, wenn die LED nicht leuchtet, wäre auch der Filter aus.

Wie bereits erwähnt, ist dies nur bei einigen Amigas nötig; die neuen Amiga 2000-B werden z.B. mit 1 MB Chip-RAM ausgeliefert, wobei natürlich dieser kleine Trick überflüssig wäre. Wir wollen ihn aber dennoch mit in den Source-Code aufnehmen, da sicherlich nicht alle Amigabesitzer über “Nur-Chip-RAM“ verfügen.

Hier ist nun der entsprechende Befehl um die Power-LED auszuschalten:

or.b #2,$bfe001

So, nun folgt die “Programmsteuerung“. Zuerst müssen wir Speicher reservieren lassen und die digitalen Daten hineinkopieren. Dazu schreiben wir uns später eine Routine, die wir mit

jsr speicher_an

aufrufen werden. Danach sichern wir uns den Zeiger auf die normale Interrupt-Routine. Das machen wir folgendermaßen:

move.l $6c,altirq

Diesen Zeiger benötigen wir, weil die Routine später von unserer Musikroutine wieder angesprungen werden muß. Nun sagen wir dem Rechner, daß er sich den Zeiger auf die Noten merken soll. Das geht so:

move.l #musik,zeiger

Das waren eigentlich schon die wichtigsten Werte, die wir setzen mußten. Nun brauchen wir nur noch den Zeiger auf unsere Interrupt-Routine verbiegen:

move.l #neuirq,$6c

Mit diesen wenigen Aufrufen wird unsere Routine auch schon gestartet. Eigentlich könnten wir unsere Musikroutine enden lassen, wenn die Musik einmal durchgespielt wurde, aber es geht auch anders. Man könnte ja auf den Gedanken kommen, das Musikstück per Maustaste zu stoppen. Dafür schreiben wir die folgende Schleife:

```
loop:
btst #6,$bfe001
bne loop
```

Was macht nun diese Schleife? Nun, es wird das Bit sechs der Adresse $BFE001 abgefragt. Solange es den Wert null innehat, wird die Schleife wiederholt. Falls das Bit gesetzt ist, also den Wert eins hat, wird die Schleife verlassen. Das Register $BFE001 ist übrigens für die Game-Ports zuständig. Das Bit sechs beinhaltet den Zustand der linken Maustaste der Maus im Port eins. Wird diese Taste gedrückt, so ist das Bit gesetzt. Schließlich muß nur noch die Ausgangsposition wiederhergestellt werden, damit kein Absturz auftritt. Dazu biegen wir als erstes den Zeiger auf die Interrupt-Routine wieder zurück:

move.l altirq, $6c

Nun müssen wir alle Sound-DMA's abschalten. Dazu löschen wir die unteren vier Bits:

move.l #$f, $dff096

Beim nächsten Schritt geben wir den belegten Speicher wieder frei, dazu schreiben wir uns später noch die Unterroutine speicher_aus. Diese rufen wir jedoch an dieser Stelle schon einmal mit folgendem Befehl auf:

jsr speicher_aus

Jetzt schalten wir die Power-LED wieder an, indem wir das Bit eins des Registers $BFE001 löschen...

bclr #1, $bfe001

Um das Programm wieder zu verlassen, geben wir den Befehl

rts

ein. Das Wichtigste an unserem kleinen Programm ist auf jeden Fall die neue Interrupt-Routine. Fangen wir also zuerst damit an:

Am Anfang stand das Label NEUIRQ

Im Leben braucht alles einen Namen, so auch unsere Routine. O.K, es ist nicht gerade umwerfend einfallsreich, sie NEUIRQ (NEUer InterRupt) zu nennen, aber diese Bezeichnung hört sich zumindest sehr viel sinnvoller an, als beispielsweise HOSENTRAEGER. Bleiben wir lieber bei dem Namen NEUIRQ. Im Listing sähe dies so aus:

neuirq:

Zuerst müssen die Register gerettet werden, da sonst der böse Guru mit einer Meditation zuschlägt, und das wollen wir doch vermeiden, oder? Der geeignetste Platz für unsere Rettungsaktion ist der Datenkeller (in der Computerbranche spricht man bei diesem Zwischenspeicher auch von dem sogenannten Stack). Der Zeiger auf dem Stack beinhaltet das Register A7. Dieses Register müssen wir retten, weil wir es später nicht überschreiben dürfen. Jedenfalls brauchen wir nur die Adreß-Register A0 bis A6 sowie die Datenregister D0 bis D7 “umzuspeichern“; und das geht so:

movem.l a0-a6/d0-d7,-(a7)

MOVEM ist ein MULTIPLE-MOVE, das heißt, daß damit gleich mehrere Register übertragen werden können. So, damit hätten wir also die Werte vor den unanständigen Veränderungen, die später noch auf die Register herniederprasseln werden, gerettet. Grund genug, nun endlich in unserer Routine fortzufahren. Der Vertical-Blank-Interrupt,

```
Wenn D5 = 0, A2 = $DFF000
     D5 = 1, A2 = $DFF010
     D5 = 2, A2 = $DFF010
     D5 = 3, A2 = $DFF010
```

den wir ja schließlich benutzen, hat die Angewohnheit, 50mal pro Sekunde aufgerufen zu werden. Nun, es ist nicht ganz so günstig für unsere Routine, wenn sie 50 Noten in der Sekunde spielen würde. Um es erst gar nicht soweit kommen zu lassen, brauchen wir eine Verzögerung in unserer Routine. Dies gelingt uns mit einem Zähler, den wir herunterzählen lassen (sonst wäre es ja auch kein Zähler!). Diesen Zähler brauchen wir immer nur mit den Wert null zu vergleichen. Wenn dieser Wert endlich null ist, dann können wir einen Ton ausgeben. So ein Zufall, daß der Wert schon gleich zu Anfang null ist. Da wird dann auch schon beim ersten Aufruf der Routine ein Ton gespielt. Hier der Vergleich mit dem Zähler...

cmp.l #0,zaehler

Wie bereits erwähnt, wenn der Zähler null ist, dann soll das Programm einen Ton ausgeben. Es verzweigt dann wie folgt...

beq spiel_musik

Nun, wenn der Zähler noch nicht null ist, dann müssen wir ihn um eins erniedrigen.

sub.l #1,zaehler

Gesagt, getan, doch: "Was nun?", sprach Zeus. Jetzt braucht nur noch die Lautstärke gesetzt zu werden. Da dieser Abschnitt auch von der Tonausgabe gebraucht wird, benötigen wir ein Label. Na gut, wir werden es liebevoll MUSIK_FERTIG nennen.

Der Ton ist's, der die Musik erst schön gestaltet

Aus diesem Namen läßt sich recht unschwer erkennen, daß hier gleich die Ausgabe der Musik fertig ist.

musik_fertig:

Doch mit diesem Label ist es leider nicht getan, denn die Lautstärke muß ja erst einmal gesetzt werden. Das geht so:

jsr volume

Das wär's dann auch schon. Nun können wir die Register-Werte wieder zurückholen. Die brauchen ja jetzt keine Angst mehr zu haben, daß sie überschrieben werden.

movem.l (a7)+,a0-a6/d0-d7

Damit wäre unsere Haupt-Interrupt-Routine fertig. Jetzt folgt der Sprung in die normale Interrupt-Routine. Wie machen wir das? Nun, wir legen die Adresse einfach auf den Stack. Und mit dem RTS wird diese Adresse dann auch angesprungen. Hier die zugehörigen Befehle:

move.l altirq,-(a7)
rts

Weiter geht's im Programm. Wir schreiben uns nun die Routine, die die Noten ausliest.

spiel_musik:

Als erstes setzen wir den Zähler. Nehmen wir den Wert acht; damit werden dann zirka fünf Noten pro Sekunde gespielt.

move.l #8, zaehler

Jetzt können wir uns daranmachen, die zugehörigen Noten auszulesen. Wir schreiben dafür eine recht einfache Routine. Die Daten sind später so aufgebaut, daß immer vier Noten hintereinander liegen, für jede Stimme eine Note. Eine Null wird eine Pause markieren, und ein $FF (255) wird das Ende des Stückes markieren. Als Zähler für die Kanal-Nummer benutzen wir das Register D5, das wir auf Null setzen.

move.l #0,d5

Beginnen wir mit der Schleife, die für jeden Kanal eine Note ausliest...

spiel_musik_loop:

Zuerst speichern wir den Zeiger auf die Noten nach A0.

move.l zeiger,a0

Dann erhöhen wir den Noten-Zeiger für den nächsten Schleifendurchlauf um eins.

add.l #1,zeiger

Nun beginnt die Verarbeitung der Noten. Falls die Note den Wert null beträgt, wird kein Ton ausgegeben, und das Programm soll verzweigen...

cmp.b #0,(a0)
beq musik_weiter

Wenn der Wert 255 ist, dann ist die Musik zu Ende. Soll sie wiederholt werden, muß auf ein bestimmtes Label (musik_neu) verzweigt werden.

cmp.b #255,(a0)
beq musik_neu

Nun können wir die Note von der "Spiel_Sound"-Routine bearbeiten lassen. Dafür müssen wir den Wert nach D0 schreiben und das Byte retten.

move.b (a0),d0
and.l #$ff,d0

Jetzt kann der Sound ausgegeben werden:

jsr spiel_sound

Zum Label Musik_weiter springt die Routine dann, wenn kein Ton ausgegeben werden soll, also eine Pause gemacht wird.

musik_weiter:

Da der Amiga vier Soundkanäle besitzt, müssen wir die Zähler um eins erhöhen, bis der Wert vier erreicht wird. Solange dieses nicht der Fall ist, wird die Schleife wiederholt.

add.l #1,d5
cmp.l #4,d5
bne spiel_musik_loop

Wenn alle Kanäle bearbeitet wurden, dann springt die Routine wieder zur Haupt-Interrupt-Routine zurück.

bra musik_fertig

Jetzt folgt die Routine, um ein Musikstück neu zu starten: Nennen wir sie einfach MUSIK_NEU. Sicherlich haben Sie schon erraten, daß diese Routine die Musik zu "neuem Leben" erweckt...

Danach setzen wir einfach den Zeiger auf die Noten:

musik_neu:

move.l #musik, zeiger

Tja, und dann geht's weiter in der Musik-Routine.

bra musik_fertig

Nun verewigen wir in unserem Programm noch das Label Zaehler, das wir ja bei der Interrupt-Routine brauchen, um eine kleine Verzögerung zu erreichen.

zaehler:
dc.l 0

So weit, so gut; wir erwähnten ja schon, daß wir noch eine Routine schreiben wollen, um den Speicher zu reservieren. Dies geschieht nun. Sie wird übrigens nicht nur den Speicher reservieren, sondern auch die digitalen Daten in den besagten Speicher kopieren.

speicher_an

Für das Belegen des Speichers existieren bereits komplette Routinen in der Exec-Library. Es gibt dabei zwei Möglichkeiten: Einmal kann der Speicher direkt belegt werden (also an eine bestimmte Adresse), und zum zweiten existiert die Routine ALLOCMEM, die damit beauftragt ist, freien Speicherplatz zu suchen. Die erste Möglichkeit werden wir nicht verwenden, weil man dabei sehr schnell einen Speicherbereich doppelt belegen kann. Dies ist übrigens meist die Ursache eines "Gurus".

In D0 (der Anfangsadresse des gefundenen Freiplatzes) übergeben wir die gewünschte Speichergröße,

move.l #10240,d0

Zehn kByte (10240 Bytes) dürften für unsere Verhältnisse ausreichen. Außerdem müssen wir den Speichertyp übergeben. In unserem Fall ist dies das Chip-RAM. Es besitzt die Kennung $02. Der Wert wird dann D1 übergeben.

move.l #2,d1

Das war auch schon fast alles. Jetzt transferieren wir noch die Exec-Basisadresse nach A6 und rufen die Routine auf, um den Speicher zu reservieren:

move.l 4,a6
jsr -198(a6)

Die Routine gibt uns in D0 den Zeiger auf den belegten Speicher zurück, den wir anschließend sichern.

move.l d0,speicher'

Eine Fehlerabfrage brauchen wir hier eigentlich nicht, weil garantiert noch 10 kByte irgendwo im Amiga frei zur Verfügung stehen. Nun kopieren wir noch die digitalen Daten in den Speicher. Dazu sichern wir den Zeiger in A1 und den Zeiger auf die Daten nach A0. Es folgt die Schleife, wobei wir immer ein Byte kopieren lassen.

move.l d0,a1
lea start,a0
speicher_an_loop:
move.b (a0)+,(a1)+

Danach lassen wir jeweils prüfen, ob das Ende der Daten erreicht wurde. Falls nicht, wird die Schleife nochmals durchgeackert. Wenn das Ende erreicht wird, erfolgt der Rücksprung.

cmp.l #end,a0
bne speicher_an_loop
rts

Für den Speicher-Zeiger brauchen wir natürlich auch ein Label und eine Routine, die den Speicher wieder freigibt:

speicher: dc.l 0
speicher_aus:

Als erstes übergeben wir den Zeiger auf den Speicher in A1 und die Speichergröße in D0. Dann sichern wir die Exec-Basisadresse nach A6 und rufen anschließend die Routine FreeMem auf, die den zuvor mit dem AllocMem-Befehl belegten Speicherbereich wieder freigibt. Damit wäre das Freigeben des Speichers fertig, und es erfolgt der Rücksprung.

move.l speicher,a1
move.l #10240,d0
move.l 4,a6
jsr - 210(a6)
rts

Nun vermerken wir in unserem Source-Code weitere Labels, die im folgenden nur kurz erklärt werden sollen.

ZEIGER – verweist auf die Noten

zeiger:
dc.l 0

ALTIRQ – zeigt auf den alten Interrupt

altirq:
dc.l 0

LAUTSTAERKE – enthält die Lautstärkenwerte für die vier Kanäle

lautstaerke:
dc.w 0,0,0,0

Nun schreiben wir die Routine, die den Sound ausgibt. Die Kanalnummer wird zuerst nach D1 kopiert, und dann schreiben wir die Adresse $DFF000,a2 nach A2.

spiel_sound:
move.l d5,d1
move.l #$dff000,a2

Wozu das alles, werden Sie sicher fragen? Nun, eigentlich müßten wir uns ja Tabellen für die einzelnen DMA-Register anlegen, jedoch können wir auch ein bißchen herumtricksen und damit Speicher einsparen. Grundlage dieser Idee ist, daß die DMA-Register der einzelnen Kanäle einen Abstand von 16 Bytes (Hex$10) haben. Wir brauchen also zu der entsprechenden Adresse von Kanal 0 einfach den Wert von Hex $10 hinzuzuaddieren. Daher multiplizieren wir die Kanalnummer, die sich in D1 befindet, einfach mit dem Hex-Wert $10 und addieren das Ergebnis mit A2.

mulu #$10,d1
add.l d1,a2

Somit hätten wir schon eine Grundlage für die DMA-Register, zu der wir nur noch Werte wie $A0 (für Sample-Start) zu addieren brauchen. Weil wir in unserem Programm diesmal auch mit verschiedenen Oktaven arbeiten, sichern wir die Note nach D1. Mit dem zweiten Befehl löschen wir die Oktave, damit nur die Note in D0 bleibt, worauf nur noch der Wert für die Oktave in die optimale Form gebracht werden muß.

move.l d0,d1
and.l #$f,d0
asr.l #4,d1

Nun wollen wir die Periods des Tones berechnen. Hierzu verdoppeln wir den Wert der Note, denn die Periods haben wir später als Wörter in einer Tabelle. Vorher holen wir noch den Zeiger auf die Periods-Tabelle nach A1 und deren Wert für die Note nach D0.

lea periods,a1
asl.l #1,d0
move.w (a1,d0),d0

Dann lassen wir die Oktave einstellen. Wenn sie gleich null ist, dann soll die Period nicht verändert werden. Falls die Oktave größer als fünf ist, dann soll ebenfalls nichts geändert werden; hat sie den Wert vier, dann soll die Period verkleinert werden.

cmp.b #0,d1
beq spiel_sound_weiter
cmp.b #5,d1
bpl spiel_sound_weiter
cmp.b #4,d1
beq oktave_erhoehen

So, nun lassen wir berechnen, wie oft die Sample-Period verdoppelt werden muß, um die richtige Oktave zu erreichen. Dazu erhält D2 den Wert drei. Von D2 ziehen wir nun D1 ab; wir erhalten dadurch einen Wert, der besagt, wie oft die Sample-Period verdoppelt werden muß. Wir verdoppeln D0 also um D2 mal. Weiter gehts dann mit einer Verzweigung zur Programmfortsetzung...

move.l #3,d2
sub.l d1,d2
asl.l d2,d0
bra spiel_sound_weiter

An dieser Stelle erfolgt das Label "OKTAVE_ERHOEHEN". Um den Ton eine Oktave höher spielen zu lassen, brauchen wir nur die Sample-Period halbieren.

oktave_erhoehen:
asr.l #1,d0

Da wir die Oktave schon bearbeitet haben, können wir jetzt endlich den Sound ausgeben lassen. Dies geschieht beim Label spiel_sound_weiter:
Als erstes geben wir die Startadresse des Samples an. Da wir das Sample an den Anfang des reservierten Chip-RAM kopiert haben, brauchen wir nur den Zeiger darauf in das Register AUD?LC zu schreiben. Sie erinnern sich noch, welche Werte in das Register A2 geschrieben wurden? Falls nicht, dann gibt Ihnen Tabelle 1 darüber Auskunft.
Wenn wir mit nachstehendem Befehl die Sound-Startadresse übergeben, wird der Wert in ein bestimmtes Register (z.B. $DFF010 + $A0 = $DFF0B0) geschrieben. Je nach Inhalt von A2 können wir demnach auf alle vier Kanäle zugreifen. Als nächstes übergeben wir die Anzahl der digitalen Daten. Diese Zahl muß halbiert werden, weil der DMA diese Daten als Worte (also immer zwei Bytes) benötigt. Abschließend übergeben wir die Sample-Period und die Lautstärke. Da wir die Lautstärke bei jedem Interruptdurchgang auch verändern wollen, speichern wir diese auch ab. Dazu holen wir uns die Start-Adresse der Tabelle nach A0.

move.l speicher,$a0(a2)
move.w #end-start/2,$a4(a2)
move.w d0,$a6(a2)
move.w #64,$a8(a2)
lea lautstaerke,a0

Dann kopieren wir die Kanal-Nummer nach D0 und verdoppeln sie, da die Lautstärke immer als ein Wort, also zwei Bytes dargestellt wird. Nun können wir die Lautstärke setzen, bevor wir den Audio-Kanal aktivieren. Dazu löschen wir zuerst das Register D0. Dann setzen wir das Bit, das der Kanal-Nummer entspricht. Hinzu addieren wir den Wert $8200 und beschreiben mit D0 das DMA-Control-Register. Abschließend folgt wieder ein Rücksprung.

move.l d5,d0
mulu #2,d0
move.w #64,(a0,d0)
clr.l d0
bset d5,d0
add.l #$8200,d0
move.w d0,$dff096

Letztendlich folgt die Routine, die dann die Lautstärke der einzelnen Stimmen verändern soll. Nennen wir sie einfach VOLUME, was ja auch Lautstärke bedeutet. Wir wollen die Lautstärke der DMA-Kanäle null bis drei bearbeiten. Daher setzen wir in D3 den Kanal null als Startwert fest. Es folgt das Label der Schleife...

move.l #0,d3
volume_loop:

In dieser Schleife lassen wir eine Routine aufrufen, die die Lautstärkenwerte in die entsprechenden DMA-Register schreiben soll. Diese Routine ist auch gleichzeitig für das Verändern der Lautstärke zuständig. Danach erhöhen wir die Kanal-Nummer um eins, worauf D3 mit dem Wert vier verglichen wird. Ist dieser Wert noch nicht erreicht, dann wird die Schleife wiederholt, andererseits erfolgt der Rücksprung.

jsr setlaut
addq.l #1,d3
cmp.l #4,d3
bne volume_loop
rts

Nun folgt die Routine, die die Lautstärke verändert. Als erstes wird die Kanal-Nummer nach D2 kopiert. Dieser Wert wird dann mit zwei multipliziert, da ja die Lautstärkentabelle nur aus Worten besteht. Den Zeiger auf die eben genannte Tabelle holen wir nach A0 und können überprüfen, ob die Lautstärke dieses Kanals den Wert null hat. Ist dies nicht der Fall, so kann zur weiteren Bearbeitung des Kanals verzweigt werden, ansonsten wird er ausgeschaltet. Dies geschieht, indem wir D2 löschen, das entsprechende Bit für den Kanal in D2 setzen und den Wert in das DMACON-Register schreiben. Anschließend erfolgt dann der Rücksprung.

setlaut:
move.l d3,d2
mulu #2,d2
lea lautstaerke,a0
cmp.w #0,(a0,d2)
bne lautsetzen
clr.l d2
bset d3,d2
move.w d2,$dff096
rts

Sollte die Lautstärke ungleich null sein, dann wird das Label lautsetzen: angesprungen. Hier sichern wir zuerst einmal die Lautstärke in D0. Dann berechnen wir die Adresse des Lautstärke-Registers, setzen die Lautstärke und dezimieren sie anschließend in der Tabelle um den Wert vier. Danach erfolgt der Rücksprung.

move.w (a0,d2),d0
move.l d3,d1
mulu #$10,d1
move.l #$dff0a8,a1
add.l d1,a1
move.w d0,(a1)
sub.w #4,(a0,d2)
rts

Es folgen die digitalen Daten und die Periods-Tabelle.

start:
dc.b -120,-100,-80,-60...
end: ;Endemarkierung
periods:
dc.w 0,427,404,381...
even

Zum Schluß folgen die Noten, die in Viererblöcken aufgebaut sind. Also, vier Bytes für vier Kanäle.

musik:
dc.b $01,$21,$00,$00...

Die erste Hex-Ziffer eines Bytes steht dabei für die Oktave, die zweite für den Ton. (br)

+++ VESALIA TOP ANGEBOTE +++

3,5" Winner-Drive* superleise, abschaltbar **235,–**
Externes Amiga-Laufwerk mit durchgeführtem Bus. Nur 25 mm hoch. Neuestes CHINON-Drive (FX 354). Amigafarbenes Metallgehäuse, helle Frontblende.

3,5" Amiga-Laufwerk* **195,–**
Externes Laufwerk mit dem millionenfach bewährten CHINON FB 354 (35 mm hoch). Abschaltbar und mit durchgeführtem Bus. Metallgehäuse und Frontblende amigafarben. Test Amiga Spezial 6/88: extrem leise.

3,5" Laufwerk A 2000* intern **155,–**
Komplett mit Zubehör und Einbauanleitung.

5,25" Winner-Drive* **279,–**
Mit Chinon FZ 506, dem Marktführer bei 5,25"-Laufwerken. 40/80 Track schaltbar und abschaltbar. Mit durchgeführtem Bus. Blende und Metallgehäuse amigafarben.

Winner-Ramkarte 512 KB für A 500* **198,–**
Mit 4 × 514256 80/100 nS (z. B. Siemens) abschaltbar. Nur noch ca. 20 % Stromverbrauch, inkl. Uhr.

512 KB-Karte mit Uhr für Amiga 500* **169,–**
komplett bestückt (16 × 41256-100 nS) abschaltbar.

1,8 MB-Karte für A 500 **698,–**
komplett bestückt, abschaltbar und mit Uhr.

2 MB-Box 512 KB bestückt für A 1000 m. Bus **498,–**
2 MB-Box mit Busdurchführung für A 1000 **798,–**
2 MB-Box komplett mit Uhr und Kick. 1.3 **1148,–**
4 MB-Box mit Busdurchführung für A 1000 **1588,–**

8 MB-Karte 2 MB bestückt für A 2000 **948,–**
8 MB-Karte 4 MB bestückt für A 2000* **1680,–**
8 MB-Karte 8 MB bestückt für A 2000* **2948,–**

Pal-Genlock-Interface V 1.3 **578,–**
Kontinuierliches Überblenden von einer Bildquelle zur anderen. Integrierter PAL-RGB-Splitter und Video-Kopierschutz-Decoder. Videobandbreite 5 MHz.

Y-C Genlock-Interface **1148,–**
Fast schon ein Videostudio. RGB-Bandbreite 10 MHz. S-VHS-Anschluß. Wandler von RGB nach Pal, auch als S-VHS→RGB-Konverter einsetzbar. Ein-/Ausblenden in beide Richtungen.

Digi-Splitt **698,–**
Vollautomatischer RGB-Splitter mit Video-Color-Prozessor und RGB/FBAS-Wandler.

Digi-Splitt Junior Testsieger in Amiga 11/89 **448,–**
Vollautomatischer RGB-Splitter für De Luxe View und Digi-View.

RGB-Splitter **298,–**
Trennt das Farbvideosignal in Rot, Grün und Blau.

De Luxe View V 4.0 **398,–**
mit deutschem Handbuch. **Amiga-Test: Sehr gut**

Mausunterlage **9,50**

BIG FAT AGNUS 8372 A 1 MB adressierbares **149,–**
CHIP-RAM-MEM inkl. Einbauanleitung. Nur für Kick. 1.3

Winner-Autobootkarte V1.1 neu* **125,–**
Jetzt auch für 2 Harddisk. Echtes Autoboot sofort nach dem Einschalten. Mit FastFileSystem auch unter Kickstart 1.2. Je nach Harddisk bis ca. 500 KB/Sek. Datentransfer. Für OMTI- und Seagate-Controller nach CT-Bauanleitung z. B. A.L.F.

Folgenden Datentransfer erzielen Sie mit unserer WINNER-Autobootkarte V1.1.

20 MB 24 mS.	5520 MFM-Controller	412 KB/Sek.
30 MB 24 mS.	5528 RLL-Controller	454 KB/Sek.
40 MB 24 mS.	5528 RLL-Controller	469 KB/Sek.
47 MB 24 mS.	5528 RLL-Controller	479 KB/Sek.
63 MB 19 mS.	5528 RLL-Controller	492 KB/Sek.

Für A500/1000 (Expansionsport) **145,–**
Für alle Amigos, Promigos, Vesalia, A 500/1000 **140,–**

Winner-Autoboot-Filecard für Amiga 2000*
20 MB 389 KB/Sek. **859,–** 31 MB 454 KB/Sek. **968,–**
47 MB 469 KB/Sek. **1198,–** 63 MB 489 KB/Sek. **1388,–**
Auch mit z. B. 2 × 20 MB, 2 × 40 MB, 2 × 47 MB und 2 × 63 MB als Filecard lieferbar. Preis auf Anfrage.

Winner-Autoboot-Einbauset A 2000*
30 MB 5,25" 444 KB/Sek. **890,–**
63 MB 5,25" 472 KB/Sek. **1288,–**

Winner-Autoboot-Harddisk für A 500/1000*
Gehäuse ideal als Monitoruntersatz. Kein zusätzlicher Platzbedarf. Maße 330 mm × 330 mm × 60 mm.
20 MB 389 KB/Sek. **988,–** 31 MB 454 KB/Sek. **1088,–**
47 MB 469 KB/Sek. **1298,–** 63 MB 472 KB/Sek. **1598,–**
Alle Harddisk und Filecard bereits unter FFS formatiert und installiert mit WB 1.3d.
Natürlich auch mit zwei Harddisk lieferbar.

5520 MFM Controller **135,–** **5527 RLL** **159,–**
5528 RLL (5527 kompatibel) höherer Datentransfer **179,–**
Festplattengehäuse* **72,–**
Amigafarbenes Metallghäuse 330 × 330 × 60 mm, Bohrungen für 2× 3,5", 1× 5,25" Harddisk, Lüfter, Netzteil, Controller und LED.
Schaltnetzteil +12 V, –12 V, +5 V **138,–**
Filecard-Trägerblech* **19,–**
*** Eigenproduktion von Vesalia Computer.**

3,5" Disketten MF 2DD 10 Stück **15,90**
3,5" Disketten DD (orig. Commodore) 10 Stück **24,50**
5,25" Disketten NN MD 2D 10 Stück **5,90**
Winner-Midi A 500/1000/2000* **93,–**
Sound-Digitizer **89,–**
Schalt- und regelbar, mit Software.

Boot-Strap* **468,–**
Nun heißt es für Amiga 500 und Amiga 2000B-User: *Bitte Kickstartdiskette einlegen.* Umschaltbar von orig. Rom auf Diskettenversion. Alles steckbar, kein Löten erforderlich. Komplett bestückt.

Elektronischer Bootselector DFO-DF3* **48,–**
Nun kann von allen Laufwerken gebootet werde. Das interne Laufwerk ist abschaltbar. Kein Löten erforderlich, nur einstecken.

ROM-ROM Umschaltplatine* **39,–**
ROM-ROM Umschaltplatine mit 1.2 oder 1.3* **99,–**
Kickstart-Umschaltplatine* **59,–**
Für 1 original Rom und 2 Epromversionen.

Händleranfragen erwünscht, kein Ladenverkauf, nur Versand

Vesalia Computer Magdalenenweg 4, 4230 Wesel · Tel. 0281/65466 · Fax 0281/64066

Listings

```
; Seka-Source-Code 4

        or.b      #2,$bfe001         ; LED ausschalten
        jsr       speicher_an        ; Speicher reservieren
        move.l    $6c,altirq         ; Alten Interrupt-Zeiger sichern
        move.l    #musik,zeiger      ; Musik-Zeiger setzen
        move.l    #neuirq,$6c        ; Neuen Interrupt-Zeiger setzen
loop:                                ; Warte-Schleife
        btst      #6,$bfe001         ; Maus-Taste gedrückt?
        bne       loop               ; Wenn nein, dann Schleife
        move.l    altirq,$6c         ; Alten Interrupt-Zeiger setzen
        move.l    #$f,$dff096        ; Alle Sound-DMAs ausschalten
        jsr       speicher_aus       ; Speicher wieder freigeben
        bclr      #1,$bfe001         ; LED einschalten
        rts                          ; Fertig
neuirq:                              ; Neue Interrupt-Routine
        movem.l   a0-a6/d0-d7,-(a7)  ; Register auf Stack ablegen
        cmp.l     #0,zaehler         ; Zähler = 0 ?
        beq       spiel_musik        ; Ja, dann spiele Noten
        sub.l     #1,zaehler         ; Zähler -1
musik_fertig:                        ; Wird später angesprungen
        jsr       volume             ; Lautstärke setzen
        movem.l   (a7)+,a0-a6/d0-d7  ; Register vom Stack holen
        move.l    altirq,-(a7)       ; Alte Interrupt-Adresse auf Stack legen
        rts                          ; Fertig
spiel_musik:                         ; Noten spielen
        move.l    #8,zaehler         ; Zähler = 8
        move.l    #0,d5              ; D5 = 0 (= Stimme-Nr.)
spiel_musik_loop:                    ; Schleife
        move.l    zeiger,a0          ; Zeiger auf Noten nach A0
        add.l     #1,zeiger          ; Zeiger um 1 erhöhen
        cmp.b     #0,(a0)            ; Note = 0?
        beq       musik_weiter       ; Wenn ja, verzweige
        cmp.b     #255,(a0)          ; Note = $FF (=255)?
        beq       musik_neu          ; Wenn ja, dann verzweige
        move.b    (a0),d0            ; Note nach D0
        and.l     #$ff,d0            ; Byte retten
        jsr       spiel_sound        ; Sample spielen
musik_weiter:                        ; Weiter geht's
        add.l     #1,d5              ; D5+1 (= Stimme+1)
        cmp.l     #4,d5              ; D5 = 4 (Alle vier Stimmen)?
        bne       spiel_musik_loop   ; Wenn nein, dann verzweige
        bra       musik_fertig       ; Verzweige
musik_neu:                           ; Musik Neu-Start
        move.l    #musik,zeiger      ; Zeiger auf Noten setzen
        bra       musik_fertig       ; Verzweige
zaehler:                             ; Für Abspielgeschwindigkeit
        dc.l      0                  ; 1 Langwort
speicher_an:                         ; Speicher reservieren
        move.l    #10240,d0          ; 10 K-Byte
        move.l    #2,d1              ; Bedingung: 2 = Chip-RAM
        move.l    4,a6               ; Exec-Basis-adresse nach A6
        jsr       -198(a6)           ; AllocMem, Speicher reservieren
        move.l    d0,speicher        ; Adresse merken
        move.l    d0,a1              ; Adresse nach A1
        lea       start,a0           ; Sound-Start nach A0
speicher_an_loop:                    ; Schleife
```


Spiele für Amiga

Neu! GREAT COURTS Neu!

DIE Tennis-Simulation für alle Amiga!
74,80 DM

Wir bekommen täglich Neuheiten für den Amiga!
Fordern Sie doch einfach mal die neueste Preisliste an,
Sie werden überrascht sein!
Hier ein paar Beispiele:

TITEL	PREIS	TITEL	PREIS
Rock'n Roll	69,90	Gran Prix Circuit	74,80
Xenophobe	74,80	Hillsfar	74,80
SimCity (dtsch)	89,90	King Arthur	74,80
Batman	74,80	Laser Squad	59,90
Chinese Karate	59,90	LittleComp.People	34,80
Create-A-Shape	98,00	Paper Boy	59,90
FS II-Hawaiien Od.	44,80	Space Quest III	99,00

COMPY SHOP

Gneisenaustr. 29
4330 Mülheim/Ruhr
02 08-49 71 69 + 49 61 78

2 MB-Rambox A1000 mit Bus 2 MB bestückt	**DM**	**799,–**
Profex 2-MB-Rambox vollbest. mit Bus A 500	**DM**	**749,–**
Amstrad LQ 3500 Letter-Quality-24-Nad.-Drucker	**DM**	**599,–**
Aztec C Professional System V3.6	**DM**	**279,–**
Digi View Gold V3.0 für A500/2000	**DM**	**279,–**
Golem 3.5-Zoll-Laufwerk	**DM**	**249,–**
Vortex 20 MB-Festplatte A500/1000	**DM**	**899,–**
Vortex 60 MB-Festplatte A500/1000	**DM**	**1499,–**
TDI-Modula Deveolpers Version V3.01	**DM**	**199,–**
MCC-Assembler+Shell+Toolkit komplett	**DM**	**99,–**
Amiga 2000 8 MB-Karte mit 2 MB bestückt	**DM**	**949,–**
Amiga 500 1.8 MB Erweiterung intern/Uhr	**DM**	**699,–**
Balance of Power / The Pawn / Thexterje	**DM**	**49,95**
BTX/VTX-Manager V2.2/BTX-Decoder dt.	**DM**	**219,–**

Kostenlose Prospkete auch für ST und IBM von

CWTG

C W T G Joachim Tiede
Bergstr. 13 * * * 7109 Roigheim
Tel./BTX 0 62 98/30 98 von 17-19 Uhr

Ist Ihr Programm der HIT ?

Der DMV Verlag sucht ständig nach neuer, interessanter Software zur Aufnahme in unser Softwaresortiment.

Dabei ist es einerlei, ob Sie nun ein Anwendungs- oder ein Spielprogramm geschrieben haben.

Der DMV Verlag bietet Ihnen sein Software-Know-How an!

```
         move.b   (a0)+,(a1)+          ; 1 Byte kopieren,
                                         Adressen+1
         cmp.l    #end,a0              ; A0 = END?
                                         Ende erreicht?
         bne      speicher_an_loop     ; Wenn nein,
                                         dann verzweige
         rts                           ; Fertig
speicher:                              ; Zeiger auf be-
                                         legten Speicher
         dc.l     0                    ; 1 Langwort
speicher_aus:                          ; Speicher frei-
                                         geben
         move.l   speicher,a1          ; Zeiger auf
                                         Speicher nach A1
         move.l   #10240,d0            ; 10 K-Byte
         move.l   4,a6                 ; Exec-Basis-
                                         adresse nach A6
         jsr      -210(a6)             ; FreeMem, Speicher
                                         freigeben
         rts                           ; Fertig
zeiger:                                ; Zeiger auf Noten
         dc.l     0                    ; 1 Langwort
altirq:                                ; Zeiger auf alten
                                         Interrupt
         dc.l     0                    ; 1 Langwort
lautstaerke:                           ; Werte für Laut-
                                         stärke
         dc.w     0,0,0,0              ; 4 Worte
spiel_sound:                           ; Ton ausgeben
         move.l   d5,d1                ; D5 (Stimme-Nr.)
                                         nach D1
         move.l   #$dff000,a2          ; Basisadresse
                                         nach A2
         mulu     #$10,d1              ; D1 * 16
         add.l    d1,a2                ; D1 zu A2 addieren
         move.l   d0,d1                ; Note nach D1
         and.l    #$f,d0               ; Note sichern
         asr.l    #4,d1                ; Oktave berechnen
         lea      periods,a1           ; Zeiger, Periods-
                                         Liste nach A1
         asl.l    #1,d0                ; Note * 2, Period=
                                         Wortbreite
         move.w   (a1,d0),d0           ; Periodwert aus
                                         Liste nach D0
         cmp.b    #0,d1                ; Oktave = 0 ?
         beq      spiel_sound_weiter   ; Wenn ja, dann
                                         verzweige
         cmp.b    #5,d1                ; Oktave => 0?
         bpl      spiel_sound_weiter   ; Wenn ja, dann
                                         verzweige
         cmp.b    #4,d1                ; Oktave = 4?
         beq      oktave_erhoehen      ; Wenn ja, dann
                                         verzweige
         move.l   #3,d2                ; D2 = 3
         sub.l    d1,d2                ; D2 = D2 - D1
         asl.l    d2,d0                ; Period um D1 mal
                                       ; verdoppeln
         bra      spiel_sound_weiter   ; Verzweigen
oktave_erhoehen:                       ; Verkleinern der
                                         Period
         asr.l    #1,d0                ; Period durch 2
                                         teilen
spiel_sound_weiter:                    ; Weiter geht's
         move.l   speicher,$a0(a2)     ; Startadresse
                                         nach AUDOLC
         move.w   #end-start/2,$a4(a2); Anzahl der Daten
                                         nach AUDOLEN
         move.w   d0,$a6(a2)           ; Period nach
                                         AUDOPER
         move.w   #64,$a8(a2)          ; Lautstärke
                                         nach AUDOVOL
         lea      lautstaerke,a0       ; Zeiger auf Laut-
                                         stärke-Register
         move.l   d5,d0                ; Stimme nach D0
         mulu     #2,d0                ; D0 verdoppeln
         move.w   #64,(a0,d0)          ; Lautstärke sichern
         clr.l    d0                   ; D0 löschen
         bset     d5,d0                ; Bit (D5) setzen
         add.l    #$8200,d0            ; $8200 addieren
         move.w   d0,$dff096           ; Audio-DMA an-
                                         schalten
         rts                           ; Fertig
volume:                                ; Lautstärke setzen
         move.l   #0,d3                ; D3 löschen,
                                         Stimme = 0
volume_loop:                           ; Schleife
         jsr      setlaut              ; Unterroutine für
                                         Lautstärke
         addq.l   #1,d3                ; D3 +1
         cmp.l    #4,d3                ; D3 = 4?
         bne      volume_loop          ; Wenn nein, dann
                                         verzweige
         rts                           ; Fertig
setlaut:                               ; Unterroutine Laut-
                                         stärke setzen
         move.l   d3,d2                ; Stimme nach D2
         mulu     #2,d2                ; Verdoppeln
         lea      lautstaerke,a0       ; Zeiger auf Laut-
                                         stärke-Register
         cmp.w    #0,(a0,d2)           ; Lautstärke = 0?
         bne      lautsetzen           ; Wenn nein, dann
                                         bearbeiten
         clr.l    d2                   ; D2 löschen
         bset     d3,d2                ; Bit (D3) setzen
         move.w   d2,$dff096           ; DMA-Kanal aus-
                                         schalten
         rts                           ; Fertig
lautsetzen:                            ; Setzen der Laut-
                                         stärke
         move.w   (a0,d2),d0           ; Lautstärke-Wert
                                         nach D0
         move.l   d3,d1                ; Stimme nach D1
         mulu     #$10,d1              ; Mal 16
         move.l   #$dff0a8,a1          ; DMA-Lautstärke-
                                         Register nach A1
         add.l    d1,a1                ; D1 dazuaddieren
         move.w   d0,(a1)              ; Lautstärke setzen
         sub.w    #4,(a0,d2)           ; Lautstärke -4
         rts                           ; Fertig
start:                                 ; Daten, Start-
                                         markierung
         dc.b     -120,-100,-80,-60,-40,-20,0
         dc.b     60,40,60,80,100,120,100
         dc.b     80,60,40,20,0,-20,-40,-160,-40,-100
end:                                   ; Endemarkierung
periods:                               ; SamplePeriod-Liste
         dc.w     0,427,404,381,359,339,320
         dc.w     302,285,269,254,240,226,214
even
musik:                                 ; Noten
         dc.b     $01,$21,$00,$00, $03,$00,$25,$00
         dc.b     $05,$21,$00,$00, $06,$00,$25,$00
         dc.b     $08,$21,$00,$00, $08,$00,$25,$00
         dc.b     $08,$21,$00,$00, $08,$00,$25,$00
         dc.b     $0A,$23,$00,$00, $06,$00,$26,$00
         dc.b     $0d,$23,$00,$00, $0a,$00,$26,$00
         dc.b     $08,$18,$00,$00, $00,$00,$28,$00
         dc.b     $00,$18,$00,$00, $00,$00,$28,$00
         dc.b     $0A,$23,$00,$00, $06,$00,$26,$00
         dc.b     $0d,$23,$00,$00, $0a,$00,$26,$00
         dc.b     $08,$18,$00,$00, $00,$00,$28,$00
         dc.b     $00,$18,$00,$00, $00,$00,$28,$00
         dc.b     $08,$23,$00,$00, $06,$00,$26,$00
         dc.b     $06,$23,$00,$00, $06,$00,$26,$00
         dc.b     $06,$21,$00,$00, $05,$00,$25,$00
         dc.b     $05,$21,$00,$00, $05,$00,$25,$00
         dc.b     $05,$18,$00,$00, $03,$00,$23,$00
         dc.b     $05,$18,$00,$00, $03,$00,$23,$00
         dc.b     $01,$21,$00,$00, $05,$00,$25,$00
         dc.b     $08,$21,$00,$00, $00,$00,$25,$00
         dc.b     $08,$23,$00,$00, $06,$00,$26,$00
         dc.b     $06,$23,$00,$00, $06,$00,$26,$00
         dc.b     $06,$21,$00,$00, $05,$00,$25,$00
         dc.b     $05,$21,$00,$00, $05,$00,$25,$00
         dc.b     $05,$18,$00,$00, $03,$00,$23,$00
         dc.b     $05,$18,$00,$00, $03,$00,$23,$00
         dc.b     $01,$21,$00,$00, $00,$00,$00,$31
         dc.b     $00,$00,$41,$00, $00,$00,$00,$31
musik2:
         dc.b     $31,$21,$00,$00, $00,$21,$00,$00
         dc.b     $00,$1a,$00,$00, $00,$21,$00,$35
         dc.b     $00,$00,$00,$00, $35,$00,$00,$00
         dc.b     $00,$38,$00,$00, $28,$00,$00,$00
         dc.b     $31,$21,$00,$00, $00,$21,$00,$00
         dc.b     $00,$1a,$00,$00, $00,$21,$00,$35
         dc.b     $00,$00,$00,$00, $35,$00,$00,$00
         dc.b     $00,$36,$00,$00, $38,$00,$00,$00
         dc.b     $28,$18,$00,$00, $00,$18,$00,$00
         dc.b     $00,$18,$00,$00, $00,$18,$00,$1c
         dc.b     $00,$00,$00,$00, $28,$00,$00,$00
         dc.b     $00,$33,$00,$00, $18,$00,$00,$00
         dc.b     $28,$18,$00,$00, $00,$18,$00,$00
         dc.b     $00,$18,$00,$00, $00,$18,$00,$1c
         dc.b     $00,$00,$00,$00, $28,$00,$00,$00
         dc.b     $00,$33,$00,$00, $18,$00,$00,$00
         dc.b     $31,$21,$45,$00, $00,$21,$00,$00
         dc.b     $00,$1a,$00,$00, $00,$21,$00,$35
         dc.b     $00,$00,$45,$00, $35,$00,$00,$00
         dc.b     $00,$38,$00,$00, $28,$00,$00,$00
         dc.b     $31,$21,$45,$00, $00,$21,$46,$00
         dc.b     $00,$1a,$48,$00, $00,$21,$45,$35
         dc.b     $00,$00,$45,$00, $35,$00,$00,$00
         dc.b     $00,$36,$00,$00, $38,$00,$00,$00
         dc.b     $28,$18,$43,$00, $00,$18,$00,$00
         dc.b     $00,$18,$43,$00, $00,$18,$00,$1c
         dc.b     $00,$00,$00,$00, $28,$00,$43,$00
         dc.b     $00,$33,$00,$00, $18,$00,$00,$00
         dc.b     $28,$18,$3c,$00, $00,$18,$41,$00
         dc.b     $00,$18,$00,$00, $00,$18,$38,$1c
         dc.b     $00,$00,$00,$00, $28,$00,$00,$00
         dc.b     $00,$33,$00,$00, $18,$00,$00,$00
         dc.b     $31,$21,$45,$00, $00,$21,$00,$00
         dc.b     $00,$1a,$00,$00, $00,$21,$00,$35
         dc.b     $00,$00,$45,$00, $35,$00,$00,$00
         dc.b     $00,$38,$00,$00, $28,$00,$00,$00
         dc.b     $31,$21,$45,$00, $00,$21,$46,$00
         dc.b     $00,$1a,$48,$00, $00,$21,$45,$35
         dc.b     $00,$00,$45,$00, $35,$00,$00,$00
         dc.b     $00,$36,$00,$00, $38,$00,$00,$00
         dc.b     $28,$18,$43,$00, $00,$18,$00,$00
         dc.b     $00,$18,$43,$00, $00,$18,$00,$1c
         dc.b     $00,$00,$00,$00, $28,$00,$43,$00
         dc.b     $00,$33,$00,$00, $18,$00,$00,$00
         dc.b     $28,$18,$3c,$00, $00,$18,$41,$00
         dc.b     $00,$18,$00,$00, $00,$18,$43,$1c
         dc.b     $00,$00,$00,$00, $00,$00,$00,$00
         dc.b     $00,$00,$00,$00, $00,$00,$00,$00
         dc.b     $ff                      ; Ende-Markierung
```

Sound-Editor

Oliver Heggelbach

Abtipphilfe

HexType V1.0 – Schnelle Eingabe von Listings

Dieses Programm dient zur Eingabe von Programm- oder Grafikdateien, die als Hexdumps (die beliebten Zahlenkolonnen) abgedruckt wurden.

In unserem Heft werden Sie öfters auf sogenannte HEXDUMP-Dateien stoßen. Dabei handelt es sich in den meisten Fällen um Programme in Assembler, C oder Modula2, deren Quellcode zu umfangreich für einen Abdruck wäre. Das vorliegende Programm macht aus diesen HEXDUMPS wieder vollständige Dateien, die je nach Beschaffenheit sofort lauffähig oder weiterverwendbar sind. Nach dem Starten des Programms unter AmigaBASIC stehen verschiedene Menüpunkte zur Verfügung.

(jb)

Wichtig !!!
Die Zeilennummern am Anfang der Programmzeilen dienen nur zur besseren Übersicht und sollten im BASIC-Editor nicht mit eingegeben werden. Sie können zur Eingabe auch den Checksummer benutzen, der sich ebenfalls in diesem Heft befindet.

Listings



```
***********************************************************
*                     AMIGA DOS INFO                      *
*Programmname   : HEXTYPE                                 *
*Funktion       : Eingabeprogramm für HEXDUMPS            *
*Sprache        : AmigaBASIC                              *
*Besonderheiten : Erlaubt das Zwischenspeichern der Files!*
***********************************************************

000:543 REM
001:784 REM HexType v1.0
002:543 REM
003:052 REM Autor: Oliver Heggelbacher
004:543 REM
005:000
006:590 CLEAR,80000&
007:416 DEFINT a-z
008:000
009:977 REM Bytes pro Zeile
010:828 bmax=16
011:000
012:984 DIM hd$(999):DIM SHARED bg,nlin
013:362 bg=0:fi=0:nlin=0
014:377 cu$=CHR$(32)+CHR$(8)
015:052 DEF FN dstell$(n$)=STRING$(3-LEN(n$),"0")+n$
016:000
017:714 WINDOW 2,"HexType v1.0"
018:822 MENU 1,0,1,"Projekt"
019:736 MENU 1,1,1,"Neues Listing"
020:070 MENU 1,2,1,"Listing speichern"
021:330 MENU 1,3,1,"Listing laden"
022:290 MENU 1,4,1,"Datei erstellen"
023:698 MENU 1,5,1,"Programm beenden"
024:174 MENU 2,0,1,""
025:302 MENU 3,0,1,""
026:430 MENU 4,0,1,""
027:382 ON MENU GOSUB Projekt
028:768 MENU ON
029:000
030:012 Warte:
031:204 WHILE NOT bg:SLEEP:WEND
032:852 WHILE fi:SLEEP:WEND
033:000
034:116 Eingabe:
035:871 PRINT FN dstell$(MID$(STR$(ze),2));":";
036:522 nlin=0:tl=0:k$="":ip$=""
037:274 Eingabe2:
038:568 WHILE k$<>CHR$(13) AND NOT nlin
039:676   COLOR 0,3:PRINT cu$;:COLOR 1,0
040:430   SLEEP
041:495   PRINT cu$;
042:386   k$=INKEY$
043:872   IF k$=CHR$(8) AND tl>0 THEN
044:057     tl=tl-1:ip$=LEFT$(ip$,tl)
045:039     PRINT CHR$(8);cu$;
046:216   END IF
047:320   IF k$<CHR$(127) AND k$>CHR$(31) AND tl<bmax*3+3 TH
        EN
048:982     tl=tl+1:ip$=ip$+k$
049:175     PRINT k$;
050:216   END IF
051:196 WEND
052:468 PRINT
053:741 IF nlin THEN Eingabe
054:448 IF LEN(ip$)<bmax*3+3 THEN
055:194   PRINT "Eingabefehler"
056:797   GOTO Eingabe
057:216 END IF
058:000
059:232 REM Checksumme ueberpruefen
060:258 ch=ze:hd$(ze)=""
061:515 FOR n=0 TO bmax-1
062:653   w$=MID$(ip$,n*3+1,2)
063:110   hd$(ze)=hd$(ze)+w$
064:501   IF w$=".." THEN w=256 ELSE w=VAL("&h"+w$)
065:471   ch=(ch*2+w) MOD 4095
066:398 NEXT n
067:712 IF ch<>VAL("&h"+MID$(ip$,bmax*3+1)) THEN
068:158   PRINT "Eingabefehler":BEEP
069:797   GOTO Eingabe
070:216 END IF
071:829 ze=ze+1:IF ze<maxz THEN Eingabe
072:381 PRINT "Listing vollständig":fi=-1:GOTO Warte
073:000
074:543 REM
075:815 REM  Menusteuerung
076:278 Projekt:
077:579 mx=MENU(0):my=MENU(1)
078:510 IF mx<>1 THEN RETURN
079:421 ON my GOTO NListing,Speichern,Laden,Datei,Ende
080:982 RETURN
081:000
082:464 NListing:
083:654 CALL Sicher("Neues Listing",1,n):IF n=1 THEN RETURN
084:468 PRINT
085:146 INPUT "Name  : ",lnam$
086:030 INPUT "Zeilen: ",maxz
087:456 INPUT "Laenge: ",dlen
088:641 PRINT STRING$(3*bmax+7,"-")
089:487 ze=0:bg=-1
090:982 RETURN
091:000
092:870 Speichern:
093:437 CALL Sicher("Listing speichern",0,n)
094:472 IF NOT bg THEN
095:094   PRINT "Kein Listing im Speicher":BEEP
096:982   RETURN
097:216 END IF
098:147 OPEN lnam$+".l" FOR OUTPUT AS #1
099:122 PRINT #1,lnam$
100:558 PRINT #1,maxz
101:480 PRINT #1,dlen
102:353 PRINT #1,ze
103:317 PRINT #1,fi
104:175 FOR n=0 TO ze-1
105:701   PRINT #1,hd$(n)
106:398 NEXT n
107:043 CLOSE #1
108:530 PRINT ze;"Zeilen gespeichert"
109:982 RETURN
110:000
111:506 Laden:
112:950 CALL Sicher("Listing laden",1,n):IF n THEN RETURN
113:640 INPUT "Name des Listings :",n$
114:099 OPEN n$+".l" FOR INPUT AS #1
115:010 INPUT #1,lnam$
116:502 INPUT #1,maxz
117:424 INPUT #1,dlen
118:089 INPUT #1,ze
119:053 INPUT #1,fi
120:175 FOR n=0 TO ze-1
121:477   INPUT #1,hd$(n)
122:398 NEXT n
123:043 CLOSE #1
124:641 PRINT STRING$(3*bmax+7,"-")
125:007 bg=-1
126:356 IF fi THEN PRINT "Listing vollständig"
127:982 RETURN
128:000
129:368 Datei:
130:061 CALL Sicher("Datei erstellen",0,n)
131:632 IF NOT fi THEN
132:934   PRINT "Listing nicht vollständig":BEEP
133:982   RETURN
134:216 END IF
135:467 OPEN lnam$ FOR OUTPUT AS #1
136:626 ze=0:sp=0:ln=0
137:952 WHILE ln<dlen
138:475   GOSUB GetW
139:992   IF w$=".." THEN
140:221     GOSUB GetW:by=w
141:475     GOSUB GetW
142:049     PRINT #1,STRING$(w,CHR$(by));
143:429     ln=ln+w
144:091   ELSE
145:159     PRINT #1,CHR$(w);:ln=ln+1
146:216   END IF
147:196 WEND
148:043 CLOSE #1
149:202 PRINT ln;"Bytes gespeichert"
150:982 RETURN
151:000
152:256 GetW:
153:341 w$=MID$(hd$(ze),sp*2+1,2):w=VAL("&h"+w$)
154:967 sp=sp+1
155:619 IF sp=bmax THEN sp=0:ze=ze+1
156:982 RETURN
157:000
158:196 Ende:
159:726 CALL Sicher("Programm beenden",1,n):IF n THEN RETURN
160:590 MENU OFF
161:464 WINDOW CLOSE 2
162:674 MENU RESET
163:500 END
164:000
165:021 SUB Sicher(te$,si,n) STATIC
166:599   nlin=-1
167:169   COLOR 3:PRINT CHR$(13);CHR$(13);te$:COLOR 1
168:136   IF si AND bg THEN
169:908     INPUT "vorhandenes Listing wird geloescht - Sich
        er (j/n) ";n$
170:275     IF n$="j" THEN n=0 ELSE n=1
171:216   END IF
172:568 END SUB
```

An dieser Stelle wollen wir noch einmal Grundsätzliches zum Thema Leserbriefe bringen. Diese Seite soll nämlich 'Ihre' Seite sein. Und wenn Sie Fragen oder Mitteilungen an die Redaktion der AMIGA DOS haben, ist es wichtig, daß Sie ein paar wichtige Formalien beachten.



SIE FRAGEN, WIR ANTWORTEN

Der AMIGA-DOS-Leserservice

Wer einen Computer kauft, weiß noch gar nicht, was später an Problemen auf ihn zukommt – dies gilt auch für den Amiga. Sind es nun nichtlaufende Programme, fehlende Druckertreiber, nichtverstandene CLI-Kommandos oder Hardware-Defekte, irgendwann hat man als Amiga-Besitzer die eine oder andere Frage. Wir von der AMIGA DOS wollen eine Art 'Rettungsanker' für Sie sein, denn nicht immer hat man die Antwort zur Frage schnell parat.

Fragen Sie ruhig, denn Fragen kostet höchstens eine Briefmarke oder einen Telefonanruf

Die Redaktion der AMIGA DOS steht Ihnen in zweierlei Hinsicht zur Verfügung:

1.) Sie schreiben uns einen ausführlichen **Brief mit beigefügtem Rückumschlag** über Ihr Problem, und wir versuchen, Ihnen so schnell wie möglich zu helfen. So schnell wie möglich heißt, daß wir entweder selbst die Antwort kennen und Sie Ihnen umgehend zusenden, oder (vor allem, wenn es um Anfragen zu in der AMIGA DOS erschienenen Programmen geht) wir leiten die Frage an den zuständigen freien Mitarbeiter weiter. Dies dauert in der Regel zwar etwas länger, dafür können Sie (ebenfalls in der Regel) mit einer kompetenten und ausführlichen Antwort rechnen.

Leserbriefe, die Probleme schildern, die auch andere Amiga-Besitzer interessieren können, werden auf den Leserbriefseiten mit einer ausführlichen Antwort abgedruckt, ebenso Briefe, deren Frage bei uns nicht direkt zu klären ist. Und deshalb sind die Leserbriefseiten **Ihre** Seiten, denn Sie sollen ein Forum sein für alle AMIGA-DOS-Leser und die, die es werden wollen, sozusagen ein *Treffpunkt der Amiganer*.

2.) Die zweite Möglichkeit, die Hilfe des Redaktionsteams in Anspruch zu nehmen, besteht dienstags zwischen 17 und 20 Uhr bei der **AMIGA-DOS-HOTLINE**. Hier stehen Ihnen die Redakteure Rede und Antwort – wenn es um Fragen zum Heft oder rund um den Amiga geht. Dazu gehören auch Probleme, die bei einer der Lieblingsbeschäftigungen mit dem Amiga entstehen können – den Spielen. Denn **nicht nur zu Soft- und Hardwareproblemen wollen wir helfen, auch Probleme beim einen oder anderen Spiel können hier behoben werden**.

Noch ein paar Hinweise zu beiden Methoden:

Wenn Sie einen Brief schreiben, bitte denken Sie daran, daß Sie **Ihre vollständige Adresse noch einmal in den Briefkopf schreiben**, somit ist gewährleistet, daß wir die Antwort auch ohne Briefumschlag noch an Sie richten können.

Bei telefonischen Anfragen, die Artikel oder Programme des Heftes betreffen, ist es sinnvoll, das entsprechende Heft neben sich zu legen. Angaben wie "Sie hatten da mal ein Programm, das funktioniert nicht!" helfen weder dem Leser noch dem Redakteur, und der Frust wird wohl auf beiden Seiten voll zuschlagen.

"So, der Worte sind genug gewechselt, nun laßt uns endlich Fragen hören!", könnte man jetzt sagen. Wir sagen allerdings noch eines: Scheuen Sie sich nicht, Fragen zu stellen, auch Anfänger haben eine reelle Chance, ihr Problem mit der AMIGA DOS zur Lösung zu bringen.

Wir freuen uns auf Ihren Anruf.

Ihre AMIGA-DOS-Redaktion

Die Tüftelecke

Das monatliche Gehirntraining

Auch in diesem Monat wollen wir Sie in den Genuß eines höchst interessanten Buches bringen, das Sie ein paar Stunden vom Bildschirm weglocken kann; denn der Computer ist zwar die zweitschönste Nebensache der Welt, aber eben nicht alles.

Wie gewohnt zeigen wir uns in dieser Rubrik nicht von der spendablen Seite, sondern verlangen zuerst einmal eine geistige Leistung von Ihnen. Diesmal ist ein kleines Kreuzworträtsel zu lösen, das einmal nicht nach einem schaurige Laute von sich gebenden Waldvogel mit drei Buchstaben oder gar nach einem unbestimmten Artikel fragt.
Denken Sie ein wenig um die Ecke, dann werden Sie nach erfolgreicher Beendigung des Rätsels in den umkreisten Feldern Buchstaben finden, die bei richtiger Zusammensetzung etwas ergeben, **was sogar Bilder bewegt.**
Schreiben Sie dieses Lösungswort auf eine Postkarte, und schicken Sie sie an den

DMV-Verlag
Redaktion AMIGA DOS
Stichwort "Rätsel"
Postfach 250
3440 Eschwege.

Einsendeschluß ist der 1. Februar 1990. Der Rechtsweg ist ebenso ausgeschlossen wie es die Mitarbeiter des DMV-Verlages und deren Angehörige von der Teilnahme sind.

Die Preise

Zu gewinnen gibt es zehn Exemplare des Rätselbuches "Um die Ecke gedacht" von "Rätselautor" Eckstein, erschienen im Rasch und Röhring Verlag.

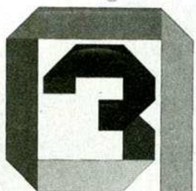
Waagerecht:

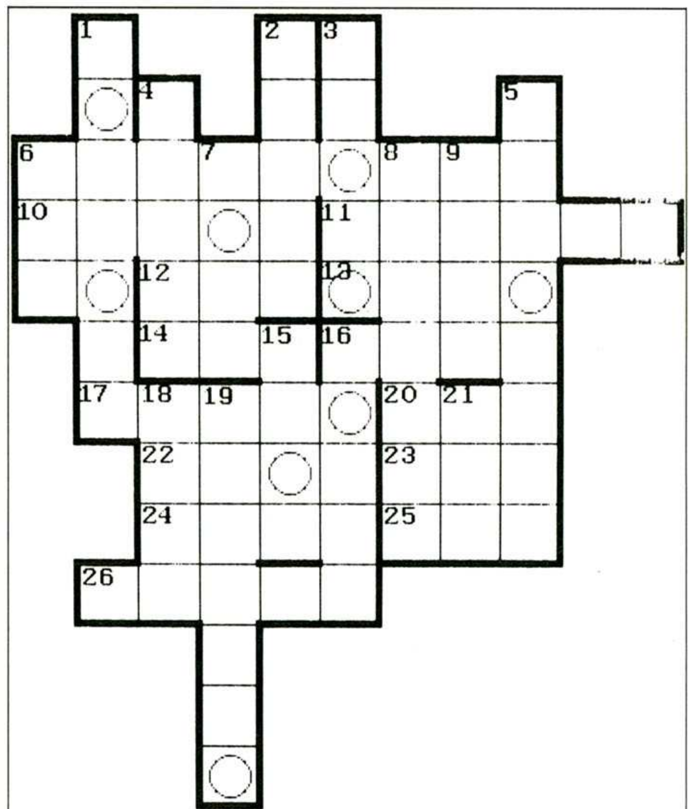
6. Wenn er Herrscher in diesem unserem Lande wäre, dann gäbe es bei der Bundeswehr nur noch Hitparaden.
10. In dieser Stadt begrüßt Hussein II. seinesgleichen.
11. Dieser "Män" hat sein fliegerisches Vorbild in Walt Disneys Dumbo.
12. Mit dem oder der Bahn fährt der Bewußte.
13. Eine mit dem Fahrzeug verbundene Lernkunst führt zur Eigenständigkeit.
14. Sie fließt durch das Charakteristikum der Bienen, nicht durch das der Deutschen.
16. Ein solch Rettendes erhofft der Ertrinkende.
17. Die zweite solche Sache wird vom Computer ausgefüllt (s.o.).
20. Nicht jeder, der in solchem sitzt, muß eine Erkältung davontragen.
22. Wer dieses mit 'h' schreibt, der lacht auch über die Zeit.
23. Lebenslängliche Verbundenheit im Stehen.
24. Streng genommen müßte es heißen: "Die Mond ist aufgegangen", nicht wahr, Frau (...)?
25. Sag's mir anders, jetzt!
26. Wer hiermit in Jugoslawien bezahlen will, der wird für einen verwirrten Römer gehalten.

Senkrecht:

1. Was Demokraten abgeben und deren Vertreter in Mengen erhoffen.
2. Werners Bölkstoff
3. Digitale spanische Freundin
4. Was dem Amiga sein Guru, das ist dem Atari seine (...).
5. Tätigkeit eines Schweizers, der mit düstrer Mine der Dinge harret, die da kommen.
6. Nur selten schreibt ein ägyptischer Gott sich so.
7. Obwohl sie solchen pflegt, wird sie als Putzfrau tituliert.
8. Hörbares Ausatmen eines vor Selbstmitleid Zergehenden.
9. Vogel, den wir in unserer Zeitung zu vermeiden gedenken.
15. Ich denke, also bin ich. Mein Denken ist mein (...).
16. roher Europäer (!)
18. Sie mit 'K' erzeugt sie mit 'B', sofern getroffen wird.
19. Wenn solcher beschmutzt wurde, dann ist das Kind bereits in selbigen hineingefallen.
21. "Welch Lüge! Hier steht er ja doch!", werden Sie empört sagen, zurecht!

(jf)





ASM-Hit

mit folgender Wertung:

Grafik	7
Sound	6
Spielablauf	10
Motivation	12
Preis/ Leistung	11

(ASM 12/89)

Einfach zu spielen, schwer zu meistern, unmöglich aufzuhören!!!

Lehnen Sie sich zurück und machen Sie es sich bequem, bevor Sie BlockOut beginnen. Sie könnten eine Weile nicht mehr von Ihrem Computer loskommen; wenn Sie einmal angefangen haben, mit den 3-D Blöcken zu spielen, hören Sie so schnell nicht wieder auf.

Das Spiel beginnt mit einem Abgrund, in den verschiedenartige geometrische Formen hinabfallen. Durch Drehen und manövrieren müssen Sie die Blöcke so anordnen, daß Sie eine Fläche bilden, die daraufhin verschwindet. Aber passen Sie auf, das Spiel wird schneller und bald schon sind die Formen so komplex, daß Sie an den Rand des Wahnsinns getrieben werden.

BlockOut fordert Ihre Reflexe und Ihren Verstand. Die größte Herausforderung aber wird es sein, vom Computer wieder wegzukommen!

- Nahezu 1000 mögliche Kombinationen von Schacht und Blöcken
- Pausenmodus
- Animierte Hilfsfunktion
- Speichern der Parameter auf Diskette
- Anfängermodus



BLOCK OUT

Erhältlich für Amiga und PC 3.5" sowie PC 5.25"

Rainbow Arts

Informationen: Rainbow Arts Software GmbH, Hansaallee 201, 4000 Düsseldorf 11 Hotline 0211-596761

Vertrieb: Rushware, Bruchweg 128, 4044 Kaarst Österreich: Karasoft Schweiz: Thali AG

DIGI VIEW GOLD

Digi View Gold und Amiga – ein leistungsstarkes Gespann

Die Preise für Hard- und Software, die dem Digitalisieren von Vorlagen dienen, sind seit ihrem Erscheinen erheblich zurückgegangen, so daß Sie nun auch für den Heimanwender mit schmalem Geldbeutel erschwinglich sind. Wer einen Amiga und einen Monitor erworben hat, bei dem ist meist Ebbe im Portemonnaie angesagt. Trotzdem wird nun noch die Portokasse geplündert, und der Anwender trägt stolz das Programmpaket Digi View Gold nach Hause. Was ist nun mit dieser Minimalkonfiguration zu erreichen?

Digi View Gold ist ein Programmpaket, das dem Digitalisieren von Vorlagen dient. Der Lieferumfang besteht einem A/D-Wandler, der dazugehörigen Steuersoftware sowie einem deutschen und einem englischen Handbuch. Weiterhin ist noch eine Farbfilterscheibe enthalten, mit deren Hilfe jeweils der Rot-, Grün- und Blauanteil einer Vorlage einzeln digitalisiert werden kann. Voraussetzung für das Arbeiten mit Digi View Gold ist ein Amigarechner mit einer Speicherkapazität von mindestens 512 kByte. Mit dieser Systemkonfiguration ist allerdings nur ein Arbeiten in der niedrigsten Auflösungsstufe möglich, werden höhere Auflösungen gewünscht, muß die Speicherkapazität den jeweiligen Bedürfnissen angepaßt werden.

Vor das Digitalisieren haben die Entwickler die Hardwareinstallation gesetzt

Bevor nun mit dem Digitalisieren begonnen werden kann, sind einige Vorarbeiten zu leisten. Dabei sammelt Digi View Gold die ersten Pluspunkte. Die Installation der Hardwarekomponenten ist wirklich einfach; zuerst wird der A/D-Wandler am Parallelport des Amiga installiert. Nun muß lediglich noch das Videosignal der Kamera an den A/D-Wandler mittels eines Chinch Kabels herangeführt werden, und der erste Teil des Hardwareaufbaus ist bewältigt. Anschließend wird die drehbare Filterscheibe so unter der Kamera montiert, daß das Objektiv jeweils von dem blauen, grünen oder roten Teil der Scheibe bedeckt ist. Ein Stativ vermag nun gute Dienste zu leisten, da bereits kleinste Verschiebungen des Aufbaus sich in verschwommenen Konturen niederschlagen. Je nach verwendeter Kamera gilt es nun, einen optimalen Objektabstand zu finden, indem das Motiv einerseits bildfüllend, andererseits auch mit der nötigen Schärfe abgebildet werden kann.

Not enough Memory, die Geißel des Digitalisierens

Bei unseren Versuchen verwendeten wir einen Amiga 500 in Normalkonfiguration, das heißt ohne zusätzliche Laufwerke, Drucker, etc. Lädt man nun nach der Workbench Digi View Gold, kann man zwischen Digi View Gold Pal sowie Digi View Gold NTSC wählen.

Bei diesem Format wird eine Auflösung von 320 * 256 geboten, die der Amiga mit 512 kByte allerdings nur ohne jegliche Peripherie laden konnte. Mangelnde Speicherkapazität war die Ursache. So kam Digi View Gold NTSC doch noch zu Ehren, die dort verwendete Auflösung von 320 * 200 ließ sich problemlos installieren. Wer allerdings über genügend Speicher verfügt, der kann mit Digi View Gold kleine Wunder vollbringen. So wird eine maximale Auflösung von bis zu 760 * 512 geboten, die die Vorlagen in fast fotografischer Qualität abbildet.

Geduld ist der Schlüssel zum Erfolg – auch beim Digitalisieren

Nachdem die Präliminarien nun weitgehend bewältigt sind, muß das Motiv entsprechend beleuchtet werden, will man eine möglichst hohe Farbechtheit erreichen. Die Stärke der Beleuchtung ist dabei zu gleichen Teilen von der Lichtempfindlichkeit der Kamera, und von dem Helligkeitsgrad der Vorlage abhängig. Um das Abbild des Mo-



tivs in den Rechner zu bekommen, muß die Vorlage nun in drei Schritten digitalisiert werden. Dabei ist es egal, in welcher Reihenfolge die einzelnen Farbauszüge aufgenommen werden. Sind alle drei Auszüge zufriedenstellend aufgenommen, wird der Menüpunkt "Display" aufgerufen, der die Auszüge zu einem kompletten Bild zusammensetzt. Dieses Rohbild kann nun als RGB-File gespeichert werden. Dies ist zwar recht speicheraufwendig, bietet jedoch den Vorteil, daß das Rohbild jederzeit wieder eingeladen und nachbearbeitet werden kann. Die Möglichkeiten, die Digi View Gold zur Nachbearbeitung zur Verfügung stellt, sind reichhaltig. Das Color Menü beinhaltet sieben Schieberegler, mit deren Hilfe einerseits die R,G,B-Anteile in ihrer Zusammensetzung in weiten Grenzen verändert werden können, andererseits können hier auch Schärfe, Kontrast, Helligkeit und Farbsättigung in weiten Grenzen eingestellt werden. Doch damit sind die Möglichkeiten des Color-Menüs bei weitem noch nicht ausgeschöpft. Digi View Gold unterstützt selbst in der Minimalkonfiguration einen 4096+-Farben-Mode, der eine äußerst naturgetreue Abbildung gewährleistet.

Wahlweise stehen, bei entsprechenden Objektfarben, noch ein 4096+-, ein 32- und ein Halfbrite-Modus zur Verfügung. Auf diese Weise kann bei Vorlagen, die aus nur wenigen Farben zusammengesetzt sind, viel Speicherplatz gespart werden. Ist das Bild fertig bearbeitet, kann es als IFF-File abgelegt werden.

Dies hat den Vorteil, daß die so erzeugten Bilder von anderen Grafikprogrammen, wie z.B. dPaint, weiterbearbeitet werden können. Hier sind natürlich Abstriche hinzunehmen; wenn ein Bild im 4096+-Farben-Modus gespeichert wurde, ist dPaint nicht in der Lage, das Bild farbecht zu reproduzieren. Dafür klappt die Zusammenarbeit mit Digi Paint III (siehe auch Testbericht zu Digi Paint III) umso besser. Ein Bild, im Modus 4096+ aufgenommen, konnte problemlos eingeladen werden und stimmte sowohl im Format als auch in der Farbwiedergabe mit dem digitalisierten überein. Wäre noch einiges zum Zeitaufwand zu sagen, der zum Digitalisieren erforderlich ist. Jeder Farbauszug benötigt etwa 20 Sekunden Rechenzeit, so daß ein komplettes Bild in zirka einer Minute erstellt ist. Hat man die optimalen Einstellungen bereits gefunden, kann man so recht zügig Bild für Bild digitalisieren. Da die Vorbereitungen innerhalb fünf Minuten zu erledigen sind, kann man Digi View Gold nur als sehr anwenderfreundlich bezeichnen.

Digi View Gold macht das Digitalisieren fast zum Kinderspiel

Das Handbuch, das dem Programmpaket beiliegt, ist übersichtlich gegliedert und in einer leicht verständlichen Form geschrieben. Jeder Menüpunkt ist sorgfältig erklärt, und läßt kaum Fragen offen. Hier findet der Anwender auch Angaben über den benötigten Speicherraum für geplante Auflösungen. Der High-Res-Modus mit vertikalem und horizontalem Overscan beispielsweise benötigt 2,5 MByte Arbeitsspeicher, um arbeiten zu können. Die Ergebnisse, die sich so erzielen lassen, lassen den Anwender schnell den Aderlaß für eine Speichererweiterung verschmerzen, die die Voraussetzung für professionelles Arbeiten mit Digi View Gold ist. Die Software ist komplett menügesteuert, so daß nicht ständig das Handbuch zu Rate gezogen werden muß. Allerdings muß jeder Anwender erst einmal die optimale Einstellung seiner verwendeten Kamera herausfinden. Dies sollte jedoch keine ernstzunehmende Schwierigkeit darstellen. Wer sich ernsthaft mit Digitalisieren beschäftigt, der wird in Digi View Gold einen guten Partner finden. Die Möglichkeiten sind derart reichhaltig, daß man Stunden braucht, will man jeden möglichen Effekt einmal ausprobieren. Wer zum Beispiel seine eigene Slideshow realisieren will, findet in Digi View Gold eine preiswerte und qualitativ hochwertige Hard- und Software, mit deren Hilfe sich beachtliche Ergebnisse erzielen lassen.

Wer sich die Arbeit noch leichter machen will, der kann mit einem RGB-Splitter das Drehen der Filterscheibe einsparen, da der Splitter ja gleich die Rot-, Grün- und Blau-Anteile an den Amiga liefert. Vorteil: Man vermeidet die Verschiebungen, die gelegentlich auftreten, wenn die Filterscheibe nicht ganz gerade montiert wurde oder sich die Folie bereits etwas geworfen hat. Schaltungstechnisch wird der RGB-Splitter zwischen der Kamera und dem A/D-Wandler positioniert. Wir empfehlen Ihnen, bei Verwendung eines RGB-Splitters abgeschirmtes Chinchkabel zu benutzen, um Störeinflüsse auf ein möglichst geringes Maß herabzusetzen. Das abgeschirmte Kabel ist zwar etwas teurer, die Ergebnisse rechtfertigen diesen Aufwand aber auf jeden Fall.

(mm)



Bild 1: Die Ergebnisse, die mit Digi View Gold zu erzielen sind, können sich sehen lassen und halten auch den Vergleich mit wesentlich teureren Systemen dieser Zielrichtung stand



Bild 2: Das Digi-View-Gold-Programmpaket besteht aus einem A/D-Wandler, einer Farbfilterscheibe und der benötigten Software

AMIGA DOS Blitzlicht

Name: Digi View Gold

Funktion: Digitalisieren von beliebigen Vorlagen

Hersteller: NewTek Inc.

Anbieter: Casablanca

Preis: 298,– DM

Positiv:
- einfache Bedienung
- übersichtliches Handbuch
- qualitativ hochwertige Ergebnisse
- Möglichkeit zur Nachbearbeitung
- attraktiver Preis
- sinnvolles Zubehör wird angeboten

Negativ:
- 1 MByte zum sinnvollen Arbeiten erforderlich
- Filterscheibe könnte robuster sein

Kompakter Schönschreiber

Der 24-Nadel-Drucker EPSON LQ-400 im Test

Wenn man von Druckern redet, kommt man unweigerlich auf die Firma EPSON zu sprechen, vor allem, wenn es um die große Frage der Kompatibilität geht. Aber EPSON hat nicht nur den Standard gebracht, auch für ihre Drucker ist die Firma bekannt. Ein Beispiel unter vielen: der LQ-400.

So ein Amiga ist ja ein herrliches Gerät, er kann Musik machen, und er kann zeichnen, und, oh Wunder, er kann auch Texte bearbeiten. Man sollte es kaum glauben, aber auch hierzu ist unser Lieblingscomputer zu gebrauchen. Leider kann er aber eines nicht, nämlich drucken. Und hier kommt nun EPSONs kleiner Druckgigant ins Spiel.

Nicht groß in den Ausmaßen – aber groß beim Drucken

Der LQ-400 ist eigentlich ein etwas abgespeckter ehemaliger LQ-500, der wiederum eine Namenswandlung zum LQ-550 und eine etwas aufstockende Verwandlung mitmachte. Somit kann man den 400 ruhig als Neuheit in EPSONs Druckerpalette vorstellen.

Der 24-Nadler besitzt zwei integrierte LQ-Schriftarten: Roman und Sans Serif. Außerdem sind optional sieben zusätzliche LQ-Schriftarten auf einem Modul erhältlich. Zu diesem Zweck befindet sich an der Seite des Druckers ein Modulschacht.

Insgesamt 14 internationale Zeichensätze lassen sich durch DIP-Schalter einstellen, außerdem auch die Zeichensätze Kursiv, Legal und der erweiterte EPSON-Grafik-Zeichensatz. Der Befehlssatz ist kompatibel zum EPSON ESC/P-Befehlssatz, Anpassungen an andere EPSON LQ-Drucker sind somit kein Problem.

Zur Ausstattung des LQ-400 gehören außer dem natürlich notwendigen Farbband und dem Netzkabel noch eine Einzelblattzuführung mit dazugehörender Druckkopfabdeckung. Eine automatische Einzelblatteinführung ist ebenfalls optional erhältlich. Soviel zur Pflicht, kommen wir jetzt zur Kür, bei der unser Testkandidat zeigen soll, was er kann.

Druckertests haben im allgemeinen den Nachteil, zwar ausführlich über das Testgerät zu berichten, die Belange eines Computerbesitzers jedoch hintenanzustellen. Deshalb will ich diesen Test aus der Sicht des Benutzers beschreiben.

Vom Auspacken zum Inbetriebnehmen in drei Minuten

Ein Drucker am Amiga muß mehrere Voraussetzungen erfüllen: Er muß Text einwandfrei und schnell ausdrucken, und er muß Grafiken in sauberer Qualität erstellen. Um es vorwegzunehmen, beim LQ-400 trifft beides zu. Kein Kunststück, wenn man 24 Nadeln besitzt. Schon der Anschluß am Amiga 500 gestaltete sich problemlos. Unser LQ wurde mit einem schon auf deutsche Bedingungen voreingestellten DIP-Schalter geliefert, man kann ruhig davon ausgehen, daß dies auch bei den anderen Geräten der Fall ist. Unter PREFERENCES der Workbench 1.2 wurde danach der LQ-Modus ausgewählt und gleich im CLI ausprobiert. Die Ausgabe von ASCII-Dateien auf das Device PRT: funktioniert ohne Probleme, der LQ verrichtet diesen ersten Teil seiner zukünftigen Arbeit reibungslos.

Zweiter Versuch: Unter der Workbench 1.3 wird in Preferences ebenfalls der Drucker eingestellt, und zwar mit der Einstellung GENERIC. Ohne lange Umschweife das Ergebnis: Es entsprach voll den Erwartungen. Nun ging es ans Eingemachte, mehrere Textverarbeitungen wurden zum Test hinzugezogen. Texte wurden mit mehreren Schriftarten wie kursiv, fett, breit, hoch- und tiefgestellt bestückt. Auch hier vermochten wir keine Beeinträchtigung zu finden. Dank des erweiterten Grafikzeichensatzes wurden alle deutschen Sonderzeichen ohne Probleme ausgedruckt.

Das gleiche trifft für Grafiken zu, unter Deluxe Paint erstellte Bilder wurden gut und sauber umgesetzt, das einzige, worauf man beim LQ-400 verzichten muß, ist die Farbe.

Der LQ-400 und sein Design – funktionell und doch ansprechend

Das Bedienfeld des LQ-400 ist mit Sensortasten ausgerüstet, mit diesen drei Tasten lassen sich die Schriftarten, Form-Feed, Line-Feed, On-Off-Line und zwei Selbsttestmodi einstellen. Die Tasten sind gut zu bedienen, auch wenn man bei Sensortasten immer das Gefühl hat, bei einem Druck zuviel hat der Garantieanspruch seine Berechtigung.

Der Druckkopf des 24-Nadlers ist eigentlich recht klein, der ihn umgebende Kühlkörper kommt einem ebenfalls etwas klein vor. Heiß werden sie ja alle, die Matrix-Druckköpfe, gute Kühlung ist also vonnöten. Obwohl der Druckkopf dabei scheinbar etwas schlecht wegkommt, tat der LQ-400 seinen Dienst auch im Dauertest, zu dem der Ausdruck einer 14seitigen Beschreibung eines Public-Domain-Programmes, der fünfmalige Ausdruck einer Schaltzeichnung und drei Briefe an die Verwandtschaft

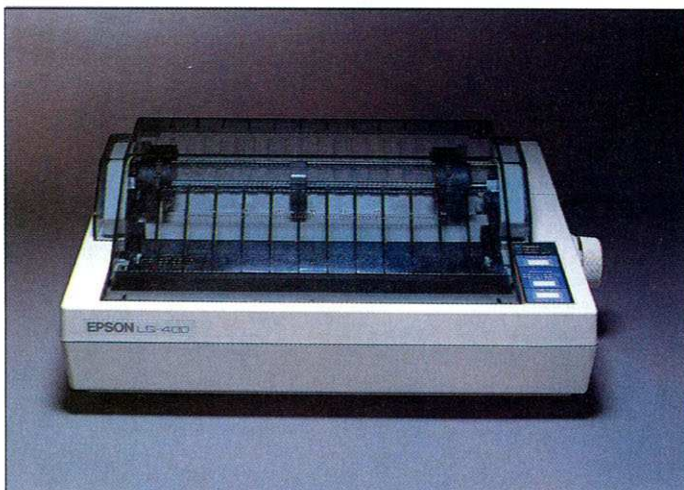


Bild 1: Der LQ-400 von EPSON ist ein 24-Nadel-Gerät. Seine Ausstattung ist recht reichhaltig



Bild 2: Der Druckkopf des LQ-400 ist kleiner, als man annehmen möchte

Der EPSON LQ-400 bietet folgende Druckmöglichkeiten :

Fettschrift, tiefgestellte oder hochgestellte Schrift.
Auch *kursive Schrift* oder doppeltbreite sowie
doppelthohe oder schmale Schrift ist möglich.
Die Schrift kann mit Umrissen, Schatten oder einer Kombination aus Umriss und Schatten ausgedruckt werden.
Der EPSON LQ-400 verfügt über die Schriftarten
ROMAN, die hier zu sehen ist und
Sans Serif, die diese Zeile bestimmt.
Unterstrichene Wörter oder Sätze sind ebenfalls kein Problem!

LQ

Abb.1: Eine Schriftprobe des LQ-400 mit einer Textverarbeitung

des Autors dieses Testes in Reihe gehörten.

Der Geräuschpegel des Druckers ist normal, zur Erleichterung nicht zu laut, wie bei Matrixdruckern aber leider üblich auch nicht gerade leise. Bei Abnahme der Abdeckung wird es reich an Geräuschen, da lohnen sich schallschluckende Unterlagen. Das Auswechseln des Zugtraktors gegen die Einzelblattablage ist ein Kinderspiel, keine halbe Minute dauert dies nach etwas Eingewöhnung.

Ein kleines Manko ist der Sitz der Netz- und der Centronics-Buchse: Nur Dank einer verlängerten Papierführung gibt es fast keine Probleme mit den Kabeln. Diese sollte man etwas zusammenbinden und seitlich am Drucker herausführen oder, noch besser, man besorgt sich gleich ein Druckerkabel mit seitlicher Einführung. Beim LQ-400 lohnt sich auch der Einsatz eines Druckerständers, vor allem dann, wenn platzmäßig nicht mehr allzuviel Freiraum existiert.

Jetzt folgt das Resümee – die Beurteilung eines Testkandidaten

Der LQ-400 ist ein Drucker wie man ihn sich wünscht: Nicht nur, daß er problemlos arbeitet, dazu schreibt er noch schön und ist auch an anderen Rechnern ohne große Umstellungen (genauer gesagt, am PC nicht eine einzige) zu betreiben. Wer ein Arbeitstier braucht, der außer zum 'Malochen' noch in der Lage ist, Briefe in schön gestalteter Schrift und Zeichnungen in guter Qualität auszudrucken, der ist mit dem LQ-400 gut bedient. Wer noch ein paar Funktionen mehr will, der sollte sich bei den größeren LQs oder Drucker anderer Firmen umsehen. Auch der Preis für diesen 24-Nadler ist ein Kaufgrund, für zirka 700 DM bekommt man eine Menge moderner Technik ins Haus.

Der LQ-400 ist auf jeden Fall ein Drucker, der Spaß macht.

(jb)

AMIGA DOS Blitzlicht

Name: LQ-400
Art: 24-Nadel-Matrixdrucker
Hersteller: EPSON
Vertrieb: Fachhandel
Preis: ca. 700 DM

Positiv:
- schönes Schriftbild
- zwei integrierte Schriftarten
- leicht überschaubares Bedienfeld
- ausführliches Handbuch
- Preis
- weitere Schriftarten als Modul erhältlich

Negativ
- Sitz der Anschlußbuchsen

Unsere Telefonnummer hat sich geändert!

0 56 51/8 09-0

Ab 1. Januar 1990 erreichen Sie den DMV-Verlag während der üblichen Bürozeiten unter der oben genannten Rufnummer.

DONAU-SOFT
24 h-Schnellversand
Ihr Amiga-PD-Partner
● ab 2,50 DM ●

Alle gängigen Serien sind lieferbar

Einzeldisk	4,50 DM
ab 10 Disk	4,— DM
ab 50 Disk	3,50 DM
ab 100 Disk	3,30 DM
ab 200 Disk	3,— DM
bei Serienabnahme:	ab 2,50 DM

Preise incl. 3,5" DD-Disks
— Mit Qualitätsgarantie —
Wir kopieren nur mit doppeltem Verify. Alle Disks sind:
— 100 % Virus- und Error frei
— etikettiert.

Leerdisketten 3,5" 2DD von
Sentinel ab 1,40 DM
Sony ab 1,85 DM

3 ausführliche Katalogdisketten mit Kurzbeschreibung aller Programme gegen **10,—** DM (V-Scheck/Briefmarken) anfordern!
gratis zu unseren Katalogen: Viruskiller, CLJ-Wizard + Turbo Backup

Das große Amiga-PD-Handbuch
Band I–IV + alle 42 Disks
+ 3 Katalogdisketten
(Einzelpreise erfragen) **325;**

Pakete für Einsteiger und Anwender (jeweils 10 Disketten)
Einsteiger 1,2; Spiele 1,2,3; Sound; Grafik; Modula II
jedes Einzelpaket 35,— DM
3 Pakete nach Wahl nur 99,— DM

Floppy 3½" int. 175,– DM
Floppy 3½" ext. } abschaltbar mit allen Extras 229,– DM
Floppy 5¼" ext. 269,– DM

+ DM 5,— bei Vorkasse, + DM 8,— bei Nachnahme
Ausland: + DM 10,— (nur Vorkasse)

MAIK HAUER
Postfach 1401, 8858 Neuburg Fax: 08431/49800
Tel.: 08431/49798 (bis 22 Uhr) BTX: *Donau-Soft #

Tino Hofstede
Computerservice
An der Windmühle 8
5010 Bergheim 5

WARRIORS, SF-Adventure	19,90
HOTELMASTER, Spiel	19,90
ENGLISCHTRAINER incl. Vokabeln	39,90
FRANZÖSICHTRAINER incl. Vokabeln	39,90
VIDEOVERWALTUNG	49,90
DISK-ETIKETTEN-Druck/Programm	19,90
LIGAVERWALTUNG (6-20 Mannschaften)	49,90
PD-DISKETTEN	ab 2,60
KATALOG (Schutzgebühr in Briefmarken)	1,00

Von 9.00 bis 23.00 UHR
AMIGA PD SO GÜNSTIG WIE NOCH NIEMALS ZUVOR
WIR KOPIEREN MIT VERIFY
ÜBER 4000 PD-DISK
Rüdiger Dombrowski
Postfach 71 04 62
2000 Hamburg 71
040/ 642 82 25
Der Versand erfolgt möglichst am Tag der Bestellung

3,5" 2DD
PD incl. Qualitätsdisk
1 - 9 a DM 2,80
10 - 79 a DM 2,00
80- a DM 1,90
+Etiketten
Serienabnahme ab 200 PD a 1,75 DM
ABO Möglichkeit!

5,25" 2D
PD incl. Qualitätsdisk
10 - 39 a DM 1,30
40 - 99 a DM 1, 15
100- a DM 1,00
Serienabnahme ab 300 PD a' 0,95 DM
ALLE UNSERE SERIEN SIND IMMER AKTUELL
EIGENE SERIE "ANTARES"

PD incl. 3,5" 2DD Markendisk. aus Belgien
1 - 9 a DM 3,00
10 - 79 a DM 2,40
80- a DM 2,30
Sensationelle Neuheiten
Serienabnahme ab 200 PD a 2,10 DM

ABSOLUT NEU: deutsche Katalogdiskette; auf dieser Diskette ist der Inhalt von 6 normalen Katalodisketten enthalten incl. ANTARES Menue 4,--DM incl. Porto nur bei Vorkasse (Briefm) Vorkasse (nur Scheck oder Überweisung kein Bargeld +Porto: 6,00DM, Nachnahme 8,00 DM incl. Porto und Verpackung

T. Schulz
Ritterstr. 2 - a
4600 Dortmund 1
0231-16 22 62

MINIGOLF: DM 29,–
Auf sechzehn abwechslungsreichen Bahnen können bis zu vier Spieler ihr golferisches Können unter Beweis stellen.

Guess It DM 19,–
Ein bis vier Mitspieler kämpfen um Worte und Punkte. Wer errät die meisten Begriffe? 2500 verschiedene Worte aus 10 verschiedenen Bereichen sorgen für anhaltenden Spielspaß.

Pro-Datei DM 19,–
Eine Dateiverwaltung, die sich Ihren persönlichen Wünschen anpaßt. Sie eignet sich sowohl für den privaten als auch den geschäftlichen Bereich und ist trotz ihres Umfangs leicht zu bedienen.

HAWK Inc. DM 19,–
Decken Sie in diesem Computerkrimi das Geheimnis auf, das die Elektronikfirma HAWK umgibt.

Alle Programme in deutscher Sprache und mit schriftlicher Anleitung. Dazu bieten wir einen günstigen Update-Service.

Vorkasse: + DM 3,– / Nachnahme + DM 6,– Ausland leider nur Vorkasse

Speichererweiterungen

Amiga 500 - 512 KB intern, abschaltbar	ab	DM	179,--
Amiga 500 - 1 MB intern, abschaltbar , mit Uhr	ab	DM	449,--
Amiga 500 - 1.5 MB intern, abschaltbar , mit Uhr	ab	DM	569,--
Amiga 500 - 2 MB intern, abschaltbar , mit Uhr	ab	DM	689,--
Amiga 1000 - 512 KB intern, abschaltbar	ab	DM	398,--
Amiga 1000 - 2 MB extern, abschaltbar	ab	DM	889,--
Amiga 2000 - 2 MB intern, abschaltbar	ab	DM	998,--
Amiga 2000 - 4 MB intern, abschaltbar	ab	DM	1.698,--
Amiga 2000 - 8 MB intern, abschaltbar	ab	DM	2.998,--

Alle Speichererweiterungen sind autokonfigurierend, abschaltbar und mit sehr schnellen RAM's (100ns und schneller) ausgerüstet!
Durch Megabit-Technologie minimaler Strombedarf.
****12 Monate Garantie****

Floppylaufwerke u. Disketten

Laufwerk 3.5" intern f. Amiga 2000	DM	198,--
Laufwerk 3.5" extern, durchgeschl. Bus, abschaltbar	DM	249,--
Laufwerk 5.25" extern, durchgeschl. Bus, abschaltbar	DM	298,--
3.5" Disketten 2DD 135 tpi 10St. /16,90 50St. /82,-- 100 St.	DM	159,--
5.25" Disketten 2D 10 St. / 6,90 50St. / 33,-- 100 St.	DM	59,--

B & S Computer-Vertriebs GmbH
Beethovenstr. 33 ; 4172 Straelen 1
Tel: 02834/1249 ; Fax: 02834/6979

DONAU-SOFT
24 h-Schnellversand

Neutrale Disketten
3,5" 2DD (100 % errorfree)

	von Sentinel	von SONY/Collossus
bis 99 Stück	1,80 DM	2,10 DM
ab 100 Stück	1,60 DM	1,95 DM
ab 500 Stück	**1,40 DM**	**1,85 DM**

Laufwerke mit allen Extras
3,5" intern .... 175,– DM
3,5" extern, abschaltbar, Busdurchführung .... 229,– DM
5,25" extern, wie 3,5" + 40/80-Trackumschaltung .... 269,– DM

Sim City	97,– DM	GFA-Basic	189,– DM
B.A.D.	77,– DM	GFA-Compiler	97,– DM
Pagestream	369,– DM	512 KB-Erw. (A500)	228,– DM
Zoetrope 1.1	189,– DM	2 MB-Erw. (A500)	698,– DM

Vorkasse: + 5,– DM, Nachnahme: + 8,– DM, Ausland: + 10,– DM

MAIK HAUER
Postfach 14 01, 8858 Neuburg
Tel.: 0 84 31/4 97 98 (bis 22 Uhr)
Fax: 0 84 31/4 98 00
BTX: *Donau-Soft #

Daten- und Organisationssysteme
Hard- und Softwarevertrieb
Ihr Amiga-Fachhändler im Bergischen Land!
Hardware, Software + Zubehör:
512 Kb Erweiterung, A-500 .... DM 199,–
3,5"-NEC-Laufwerk .... DM 239,–
Druckerkabel .... DM 17,90
5,25"-TEAC-Laufwerk .... DM 299,–
Alle Gigatron-Speichererweiterungen lieferbar
Alle Hurricane + GVP Produkte lieferbar
Diverse Hardware-Erweiterungen vorführbereit
Restbestände bis zu 50% Rabatt
z.B. Stargoose .... DM 29,90
Danger Freak .... DM 29,90
Trivial Pursuit .... DM 39,90
Willow .... DM 39,90
Daley Thompson's Olympic .... DM 39,90
The Kristal .... DM 59,90
Bomb Fusion .... DM 29,90
Denaris (wie Katakis) .... DM 39,90
u.v.a. Gesamtliste gegen DM 2,– Portoerstattung
Public Domain .... ab DM 4,–
Disketten (mit GARANTIE):
3,5" 2D .... DM 16,90   3,5" 2D farbig .... DM 19,90
Dies ist nur eine kleine Auswahl aus unserem Programm.
Über weitere Produkte (Hard- und Software) können Sie sich gerne in unserem Geschäft informieren. Wie würden uns über Ihren Besuch sehr freuen!
Öffnungszeiten (Büro + Ladengeschäft):
Mo–Fr 10.00–18.30 – Sa 10.00–14.00 – langer Sa 10.00–16.00
Sedanstraße 136 * 5600 Wuppertal 2
Tel. 0202/501500 * Martin Kramer

SPIELE für AMIGA und PC
AMIGA SPIELE AMIGA SPIELE
SIM CITY 512Kb DM 84,95 KULT DM 72.-
HARD DRIVIN 55.- North & South DM 72.-
ASTERIX DM 72.- BATMAN-MOVIE DM 72.-
Wir haben mehr als 500 Spiele vorrätig, rufen Sie an.
PC-SPIELE /CGA/HERCULES notwendig
KULT, SIM CITY, OOZE je DM 72.-
PD-SOFT Bestellannahme
☎0621-673105 NN Versand

HARDWARE
Laufwerk 3,5" intern AMIGA DM 169.-
Laufwerk 3,5" extern AMIGA DM 199.-
Laufwerk 5,25" extern AMIGA DM 269.-
Speichererweiterung für AMIGA 500
512KB mit 1MB-Chips **DM 179,00**
Leerdisketten 5,25" 2D 48tpi
100Stück DM 57,00
PD-SOFT Bestellannahme
☎0621-673105 NN-Versand

6600 3300
AMIGA MS-DOS
PUBLIC DOMAIN
Leerdisketten 3,5" (2DD) Bulkware m. Aufkleber 100Stück DM 161,00
AMIGA und MS-DOS Preise
Auf 3,5" 2DD Disketten je DM 2,10
Auf 5,25" 2D Disketten je DM 1,10
PD-SOFT Bestellannahme
☎0621-673105 NN-Versand

Ihre Hotline für AMIGA-DOS-TIP

Tel.: 089/4391087 bis -89

Fax: 089/4391080

MacSoft – *AMIGA Shop*

Public Domain

JEDE DISKETTE AUF 2 DD NUR **4,–** DM

Alle gängigen Serien!
Markendisketten 2 DD!
24-Std.-Versand-Service!
Kostenloser Umtausch defekter Disketten. Keine zusätzliche NN!
Update von Katalogdisketten!
Selber Abholen! NN gespart!
Katalog-Diskette 5,-

Telefon 0231/512603

Mo.-Fr. 16-21 • Samstag 10-16
Kasselerstr. 2a • 4600 Dortmund

Oliver Heggelbacher

CheckBAS V1.0

Der AmigaBASIC-Checksummer

In der AMIGA DOS finden Sie eine Menge an BASIC-Listings vornehmlich in Amiga-BASIC geschrieben. Um Ihnen die Eingabe etwas zu erleichtern, stellen wir Ihnen 'CheckBAS V1.0 vor. Dieses Programm dient dazu, Tippfehler beim Abtippen von BASIC-Listings aufzudecken und so die Fehlersuche zu vereinfachen.

AmigaBASIC-Listings haben den den Vorteil, sich recht gut in einer Zeitschrift abdrucken zu lassen. Leider hat der BASIC-Editor den Nachteil, daß man kaum eine Überprüfung der eingegebenen Zeile vornehmen lassen kann, es sei denn, man wechselt den Blick bei jedem Buchstaben zwischen Monitor und AmigaDOS. Zur besseren Eingabe geben wir Ihnen ein Hilfsmittel in die Hand, daß Ihnen zumindest die Arbeit erleichtert: den Amiga-BASIC-Checksummer.

Anwendung des Programms

Nachdem man ein BASIC-Listing ganz normal (das heißt ohne die zwei sechsstelligen Zahlen am Anfang einer Zeile) unter AmigaBASIC eingegeben hat, speichert man es mit SAVE "Dateiname",a (als ASCII-Version) ab. Dann startet man das 'CheckBAS'-Programm. Nach der Eingabe des Dateinamens werden jeweils 20 Zeilen der Datei angezeigt, danach wird auf einen Tastendruck gewartet. Bei der Ausgabe der Zeilen wird zuerst die Zeilennummer angezeigt, danach folgt die Prüfsumme und der Inhalt der Zeile (z.B. 001:790 PRINT "Hallo"). Nun überprüfen Sie die angezeigte Prüfsumme mit der des Listings in der Zeitschrift. Sollten diese nicht übereinstimmen, so notieren Sie sich die entsprechende Zeile, und korrigieren Sie die Zeile, nachdem Sie das gesamte Listing überprüft haben.

Hinweise

1. Um ein Programm zu überprüfen, können Sie auch folgendermaßen verfahren: Nachdem Sie das zu überprüfende Programm abgespeichert haben, lassen Sie es im Speicher stehen, starten ein zweites AmigaBASIC (wozu hat man denn Multitasking) und laden in diesem AmigaBASIC das CheckBAS-Programm. Wenn Sie nun das LIST-Window des ersten AmigaBASIC in den Vordergrund holen, können Sie beim Auftreten eines Prüfsummenfehlers die entsprechende Zeile sofort korrigieren.

2. Wollen Sie, daß weniger als 20 Zeilen auf einmal angezeigt werden, so können Sie im CheckBAS-Programm die Variable "zanz" mit einem anderen Wert belegen. Die entsprechende Stelle ist mit einer REM-Zeile markiert.

CheckBAS ist eingeschränkt auch bei anderen Listings nutzbar, wenn diese als ASCII-Files abgespeichert sind.

Hier kommt es aufs Probieren an.

(jb)

Listings

```
*************************************************************
*                     AMIGA DOS INFO                        *
*   Programmname   : CHECKBAS                               *
*   Funktion       : Checksummer für BASIC-Listings         *
*   Sprache        : AmigaBASIC                             *
*   Besonderheiten : Zeigt BASIC-Zeilen mit Checksumme an!*
*************************************************************

000:543 REM
001:808 REM CheckBAS v1.0
002:543 REM
003:052 REM Autor: Oliver Heggelbacher
004:543 REM
005:000
006:416 DEFINT a-z
007:837 DEF FN dstell$(x)=STRING$(4-LEN(STR$(x)),48)+MID$(ST
        R$(x),2)
008:000
009:008 REM Anzahl Zeilen
010:200 zanz=20
011:000
012:609 COLOR 3:PRINT "CheckBAS v1.0":COLOR 1
013:746 PRINT:INPUT "Name der zu überprüfenden Datei :",qu$
014:468 PRINT
015:000
016:090 ln=0
017:971 OPEN qu$ FOR INPUT AS #1
018:059 WHILE NOT EOF (1)
019:168   LINE INPUT #1,ip$
020:522   az=0:ch=0
021:961   FOR n=1 TO LEN(ip$)
022:257     n$=MID$(ip$,n,1):IF NOT az THEN n$=UCASE$(n$)
023:260     IF n$=CHR$(34) THEN az=NOT az
024:552     IF az OR n$<>CHR$(32) THEN ch=(ch*2+ASC(n$)) MOD
       1000
025:398   NEXT n
026:229   PRINT FNdstell$(ln);":";FNdstell$(ch);CHR$(32);
027:689   COLOR 3:PRINT ip$:COLOR 1
028:472   ln=(ln+1) MOD 1000
029:392   IF (ln MOD zanz)=0 THEN
030:371     PRINT STRING$(20,45);"Taste drücken";STRING$(20,
        45)
031:756     WHILE INKEY$="":SLEEP:WEND
032:216   END IF
033:196 WEND
034:043 CLOSE #1
035:259 PRINT STRING$(21,45);"Listingende";STRING$(21,45)
```

Listing CheckBAS



DELUXE

ARBEITEN WIE EIN PROFI

Schon nach dem Einschalten von Deluxe Paint wird der Anwender mit einer Vielzahl von Icons und Symbolen konfrontiert. Welche Funktionen sich dahinter verbergen, haben wir im letzten Teil unseres Deluxe-Paint-Kurses behandelt. Dieser zweite Teil wird sich im wesentlichen mit den Menüs der Kopfleiste beschäftigen.

Kursfahrplan 

Teil 1: Das Basismenü und seine Funktionen

Teil 2: Die Kopfleiste und die dort enthaltenen Menüs

Teil 3: Die Anwendung der in der Kopfleiste enthaltenen Funktionen

Teil 4: Animation mit dPaint III

Teil 5: Arbeitstips vom Profi

Doch zunächst wollen wir mit der Erklärung des Basismenüs und der Modifizierung der dort enthaltenen Funktionen fortfahren.

Sie erinnern sich noch? Im letzten Heft hatten wir uns genauer angesehen, was passiert, wenn ein Icon aus dem Basismenü mit der rechten Maustaste angeklickt wird. Der Anwender gelangt so in Menüs, über die er die Wirkungsweise einzelner Funktionen verändern kann. Jedoch hat nicht jede Funktion auch ein Menü, über die sie verändert werden kann. So taucht hinter Funktionen wie <füllen>, <Rechteck, gefüllt, ungefüllt>, <Kreis, gefüllt, ungefüllt>, <Vieleck, gefüllt, ungefüllt> und <Ellipse, gefüllt, ungefüllt> immer der <Fill Type Requester> auf. Hinter <Gerade Linie> und <Kurvenlinien> wiederum erscheint der <Spacing Requester>.

Beide Menüs sollen an dieser Stelle nur als Beispiel dienen, da sie bereits im ersten Teil dieses Kurses genauer erläutert wurden.

Die Icons, hinter denen sich noch Neues für uns verbirgt, sind: <Text>, <Gitter> und <Symmetrie>.

Wie Text, Gitter und Symmetriefunktion modifiziert werden können

Wird das <Text>-Icon mit der rechten Maustaste angeklickt, öffnet sich dadurch der <Choose Font Requester>. Hier können die Fonts (Schriftarten), die mit Deluxe Paint mitgeliefert werden, angesehen und ausgewählt werden. In dem großen Window auf der linken Seite des Requesters sind die einzelnen Fonts mit ihrem Namen verzeichnet. Gleich daneben können Größe und Stil der Schrift eingestellt werden. Der Anwender kann zwischen <Bold> (Fett), <Italic> (Kursiv) und <Underline> (Unterstrichen) wählen. Die <Size>-Einstellung darüber gibt Auskunft über die Größe der Schrift. Wurde die gewünschte Schrifttype ausgesucht und Stil und Größe festgelegt, kann der Benutzer sich nun genau anzeigen lassen, wie die ausgesuchte Schrift schlußendlich aus



PAINT II

sieht. Dazu braucht nur das <Show>-Gadget angeklickt zu werden. <OK> und <Cancel> bedürfen wohl keiner weiteren Erklärung.

Wird das <Gitter>-Icon mit der linken Maustaste angeklickt, öffnet sich daraufhin der <Gridding Requester>. Hier können die horizontalen und vertikalen Abstände der Gitterlinien (siehe auch Erklärung der <Gridding>-Funktion, Teil 1) eingestellt werden.

Auch die <Symmetrie>-Funktion kann manipuliert werden. Wieder mit der rechten Maustaste wird der <Symetry Requester> geöffnet. Dieser Requester ist in drei Bereiche unterteilt. Wird die <Point>-Option aktiviert, geht der folgende Symmetrieeffekt von einem zentralen Fixpunkt aus. Wird <Tiles> eingestellt, geht der Effekt von einer bestimmten Anzahl von Fixpunkten aus, die über den Monitor verteilt sind. Die horizontale bzw. vertikale Anzahl der Punkte kann im <Width>- bzw. dem <Height>-Gadget eingegeben werden.

Im zweiten Bereich des Requesters finden sich die Optionen <Cyclic>, <Mirror>, <Order> und <Place>.

Mit <Place> kann der Mittelpunkt des Symmetrieeffektes versetzt werden. Allerdings ist nur bei eingeschalteter <Point> Symmetriefunktion eine Veränderung sichtbar, auf die <Tile>-Option hat <Point> keine Auswirkungen.

<Cyclic> und <Mirror> beziehen sich auf die Art des Symmetrieeffektes. Mit diesen beiden Optionen sollten Sie am besten selbst ein wenig herumprobieren. Die Wirkungsweise dieser Optionen werden beim Ausprobieren weit besser verdeutlicht, als dies eine Erklärung tun könnte.

Die Kopfleiste und was sich dahinter verbirgt

Nach dem Basismenü, in dem sich die wesentlichen Zeichenwerkzeuge befinden, wird sich der Deluxe-Paint-Anwender mit der Kopfleiste, die sich am oberen Bildrand befindet, beschäftigen. Hier werden unter den Oberbegriffen <Picture>, <Brush>, <Mode>, <Anim>, <Effects> und <Prefs> viele verschiedene Dinge zusammengefaßt.

Beginnen wir mit <Picture>. Die beiden ersten Punkte dieses Pulldown-Menüs sind <Load> und <Save>. Werden diese Optionen angewählt, können Bilder, die gerade bearbeitet werden, abgespeichert <Save> bzw. wieder eingeladen <Load> werden.

Mit <Delete> können Sie bereits auf Diskette gespeicherte Bilder wieder entfernen. Gehen Sie mit dieser Option vorsichtig um. Wird damit eine Bilddatei auf einer Disk gelöscht, so ist sie unwiederbringlich verloren, es sei denn, Sie haben irgendwo noch eine Sicherheitskopie.

<Print> ruft das Druckermenü auf. In diesem Menü werden Sie mit einer Vielzahl von Optionen konfrontiert. Wenn Sie einen Drucker an Ihrem Amiga angeschlossen haben und in den Preferences den entsprechenden Druckertreiber eingestellt haben, können Sie nun Bilder ausdrucken.

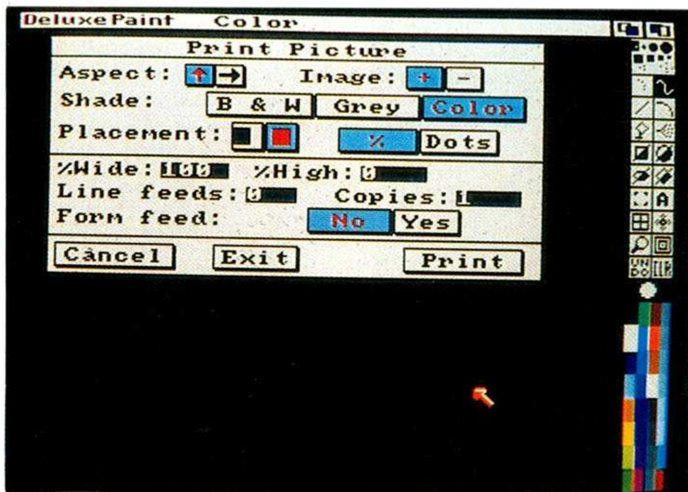
Die erste Option <Aspect> bezieht sich auf Ausrichtung des Bildes während des Druckvorgangs. Mit <Image> kann zwischen einem normalen Ausdruck und einem invertierten umgeschaltet werden. <Shade> schaltet zwischen Schwarzweiß <B&W>, Grau <Grey> und Farbausdruck <Color> um. Bedenken Sie, daß Sie die Farboption nur mit einem Farbdrucker benutzen können. <Grey> wandelt eine Farbpalette in Grauwerte um, die von einem Schwarzweiß-Drucker wiedergegeben werden kann.

<Placement> meldet Ihrem Drucker, wo Ihr Ausdruck positioniert werden soll, mitten auf dem Papier oder linksbündig.

< % / Dots>, wenn hier < % > eingestellt ist, können Sie das Seitenverhältnis in prozentualen Werten angeben, während bei der <Dot>-Option die genaue Anzahl der horizontalen bzw. vertikalen Pixel des Ausdrucks eingestellt werden können. In beiden Fällen müssen Sie entsprechende Eingaben in die Zahlfelder hinter <Dots Wide> und <High> vornehmen. <Line feeds> gibt an, wieviel Zeilenvorschub Sie nach einem Ausdruck haben möchten.

<Copies> kontrolliert die Anzahl der Ausdrucke.

Mit <Form feed> können Sie angeben, ob Sie überhaupt einen Zeilenvorschub nach dem Ausdruck haben möchten. Der Menüpunkt <Flip> offenbart, wenn er angewählt wird, ein weiteres kleines Menü, in dem die Punkte <Horiz> und



Deluxe Paints Druckermenü

<Vert> enthalten sind. Diese Option dient dazu, ganze Bilder horizontal oder vertikal zu klappen. Die beiden Kleinbuchstaben hinter den Menüpunkten beziehen sich auf die Tasten, über die die beiden Optionen auch aktiviert werden können.

Deluxe Paints bunte Seite – Farbspiele nach allen Regeln der Kunst

Der nächste Menüpunkt <Change Color> öffnet ein Untermenü, in dem alle wesentlichen Deluxe-Paint-Farboptionen enthalten sind. Die Farbpalette kann bei Deluxe Paint auf verschiedene Art und Weise geöffnet werden. Entweder über den Mausklick im <Colorindikator> oder über den Menüpunkt <Change Color> aus dem Kopfleistenmenü. Wird dieses angewählt, öffnet sich ein weiteres Untermenü, in dem Befehle, die sich unmittelbar auf Farben beziehen, zusammengefaßt wurden. Der erste Punkt in diesem Submenü ist <Palette>. Die <Palette> ist das Herzstück der Farbkontrolle. Wenn Sie die aktuellen Zeichenfarben ändern wollen, müssen Sie in das <Palette>-Menü. Aktivieren Sie das <Color Palette>-Menü, in dessen oberem Drittel sich zum einen sechs vertikale Schieberegler und zum anderen eine Darstellung der aktuellen Palette befinden. Mit den Schiebereglern kann jeweils die Farbe, die in der Palette gerade eingestellt ist, nach Belieben verändert werden. Dabei verändern die Regler mit <R>, <G> und <B> jeweils den Rot-, Grün- bzw. Blau-Farbanteil der eingestellten Farbe. Die zweiten drei Regler sind mit <H>, <S> und <V> bezeichnet. Dies steht für <Hue>, <Saturation> und <Value>. Diese Regler dienen in erster Linie dazu, neue Farben zu kreieren. Mit dem <Hue>-Regler kann die Grundfarbe eingestellt werden, der <Saturation>-Regler kontrolliert die Farbsättigung, und <Value> bestimmt den Helligkeitswert der Farbe.

Mit <Spread> kann Deluxe Paint Farbverläufe zwischen zwei Farben automatisch erzeugen. Wird <Spread> angeklickt, erscheint daraufhin unter dem Mauszeiger das Wort 'to'. Klicken Sie nun irgendeine Farbe an, und Sie werden sehen, wie Deluxe Paint zwischen der ursprünglichen Farbe und der eben angewählten einen Farbverlauf erzeugt.

<Ex> vertauscht die Werte zweier Farben bzw. deren Register.

<Copy> kopiert die Registerwerte einer Farbe über eine andere Farbe bzw. deren Register.

Die <Range>-Option wurde schon im ersten Teil unseres Kurses kurz erwähnt. Man kann mit ihr mehrere hintereinanderfolgende Farben zu einem Farbbereich verbinden, mit dem dann bestimmte Füllarten benutzt werden können (siehe Teil 1). Es können bis zu sechs verschiedene Bereiche definiert werden. In welchem Sie gerade arbeiten, können Sie an der Zahlenanzeige rechts neben <Range> ablesen. Der <Speed>-Schieberegler kontrolliert die Geschwindigkeit des Farbrollens, die hier voreingestellt werden kann.

Der Pfeil neben dem <Speed>-Regler gibt an, in welcher Richtung das Farbrollen in den aktuell eingestellten Farbbereich <Range> ablaufen soll.

<Undo> nimmt die letzte im <Color Palette>-Menü getätigte Änderung wieder zurück.

Mit <Chancel> und <OK> verlassen Sie das <Color Palette>-Menü.

Noch mehr Farbspiele – was sich im Change-Color-Menü noch verbirgt

<Use Brush Palette> – Deluxe Paint kann Bilder (Pictures) und Bildausschnitte (Brushes) abspeichern. Wenn Deluxe Paint ein Bild speichert, wählt es dazu das IFF (Interchange File Format). Dieses Format hat den Vorteil, recht universell benutzt werden zu können. Z.B. ist es ohne weiteres möglich, ein Amiga-IFF-Bild in Deluxe Paint auf einen MS-DOS-Computer zu übertragen. Der Nachteil einer IFF-Datei ist ihre enorme Länge.

Speichert Deluxe Paint allerdings einen Bildausschnitt, wählt es dafür ein völlig anderes Format. Im Grunde genommen werden Bildausschnitte wie Bilder abgespeichert, nur daß hier ein platzsparenderes Format als IFF verwendet wird.

Lädt der Benutzer einen Bildausschnitt, bleibt die aktuell eingestellte Farbpalette davon unbeeinträchtigt. Möchte der Anwender jetzt mit den Farben weiterarbeiten, aus denen sich der eben geladene Bildausschnitt zusammensetzt, kann er dieses mit <Use Brush Palette> tun. Dann werden die Farben des Bildausschnitts über die aktuellen Werte der Farbpalette kopiert.

<Restore Palette> setzt die Farbpalette wieder in den Zustand zurück, in dem sie sich unmittelbar nach Programmstart befindet.

<Default Palette>. Wenn Sie ein neues Bild eingeladen haben, aber mit der Palette, wie sie vor dem Ladevorgang aktuell war, weiterarbeiten wollen, können Sie diese mit <Default Palette> wieder herstellen.

Mit der <Cycle>-Option kann das Farbrollen aktiviert werden. Dabei werden nacheinander alle Farbregister mit allen möglichen Farben aus den anderen Registern gefüllt. Dadurch entstehen wild rotierende Farborgien. Der <Cycle>-Effekt kann auch über die TAB-Taste an- und ausgeschaltet werden.

<Bg -> Fg> tauscht die aktuelle Hintergrundfarbe eines Bildes mit der aktuellen Zeichenfarbe.

<Bg <-> Fg>. Mit dieser Option können Sie alle Pixel eines Bildes, die in der aktuellen Hintergrundfarbe gehalten sind, in die aktuelle Vordergrundfarbe konvertieren. Alle Bildteile, die in der aktuellen Vordergrundfarbe gehalten sind, werden zur Hintergrundfarbe konvertiert.

<Remap> paßt die Farben eines nachgeladenen Bildausschnittes denen des aktuellen Bildes an. Das heißt, Deluxe Paint sucht aus der aktuellen Bildpalette die Farben heraus, die den in dem Bildausschnitt verwendeten am nächsten kommen. Außerdem ist <Remap> der letzte Menüpunkt des <Change Color>-Untermenüs.

Deluxe Paints zweites Gesicht – Der Spare-Arbeitsbildschirm

Nächster Punkt des <Picture>-Menüs ist <Spare>, hinter dem sich ein weiteres Untermenü verbirgt.

Findet Deluxe Paint in Ihrem Computer genügend Speicher vor, wird neben dem eigentlichen Zeichenbildschirm noch ein zweiter Bildschirm angelegt, der aber nicht unmittelbar sichtbar ist.

Der erste Menüpunkt <Swap> dient dann auch folgerichtig dazu, zwischen den beiden Bildschirmen hin- und herzuschalten.

<Copy To Spare> kopiert das aktuelle Bild aus dem Bildschirm, der angezeigt wird, in den <Spare>-Bildschirm.

<Merge in front> kopiert den Inhalt des nicht sichtbaren <Spare>-Bildschirms in den sichtbaren Arbeitsbildschirm. <Merge in back> kopiert den Inhalt des aktuellen Zeichenbildschirms über den Inhalt des <Spare>-

Bildschirms. Die beiden <Merge>-Optionen verbinden also jeweils die Inhalte des Arbeitsbildschirms und des <Spare>-Bildschirms. Mit <Delete this Page> kann der Inhalt des aktuell sichtbaren Bildschirms gelöscht werden, ohne Einfluß auf den nicht sichtbaren <Spare>-Bildschirm zu nehmen. <Delete this Page> ist außerdem der letzte Menüpunkt des <Spare>-Untermenüs.

Nächster Punkt des <Picture>-Menüs ist <Page Size>. Hier kann zwischen drei voreingestellten Formaten für den Zeichenbildschirm und einer numerischen Eingabe des gewünschten Formates umgeschaltet werden. Hier ist anzumerken, daß dem Bildschirmformat eigentlich nur durch den Speicherplatz Grenzen gesetzt sind.

Haben Sie für Ihr Bild ein Format gewählt, das größer ist als der Bildschirm, können Sie sich mit <Show Page> das ganze Bild in einem verkleinerten Maßstab darstellen lassen.

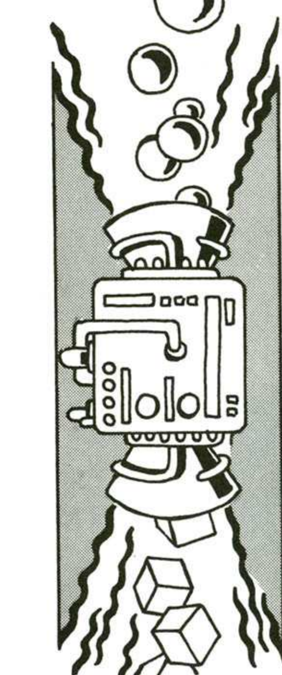
Der Menüpunkt <Screen Format> öffnet das <Choose Screen Format>-Menü, das Sie schon während des Ladevorgangs von Deluxe Paint passieren mußten. Hier kann während der Arbeit mit Deluxe Paint das Bildschirmformat und der Mode, in dem gearbeitet werden soll, verändert werden. <About> gibt Auskunft über die Versionsnummer des Programms und enthält noch ein paar Informationen über den zur Verfügung stehenden Speicherplatz.

Mit <Quit> kann Deluxe Paint zur Workbench verlassen werden.

Bildausschnitte grenzenlos – Das Brush-Menü im Detail

Das zweite Menü der Kopfleiste ist <Brush Menü>. Hier sind alle wesentlichen Deluxe Paint Optionen enthalten, die sich auf die Arbeit mit Bildausschnitten (Brushes) beziehen.

<Load> und <Save> entsprechen der Funktion der <Load>- und <Save>-Option, wie sie auch im <Picture Menü> auftaucht. Auch die Arbeitsweise entspricht dem weiter oben beschriebenen Vorgang.



Im <Brush Menü> gibt es viele Möglichkeiten, Form und Farbe eines aktuellen Bildausschnittes verändern zu lassen. Der dritte Menüpunkt des <Brush Menüs>, <Restore>, dient dazu, getätigte Veränderungen wieder rückgängig zu machen.

Der vierte Punkt des <Brush Menüs> öffnet ein weiteres Untermenü, in dem Befehle zusammengefaßt wurden, die in Zusammenhang mit der Größe eines Bildausschnittes stehen, mit dem gerade gearbeitet wird.

Mit <Stretch> kann ein Bildausschnitt in jede beliebige Größe gedehnt oder zusammengeschoben werden.

<Halve> halbiert einen Bildausschnitt sowohl in seiner horizontalen als auch in seiner vertikalen Abmessung.

<Double> verdoppelt die horizontale und vertikale Größe eines Bildausschnitts <Double Horiz> und <Double Vert> verdoppeln jeweils nur Höhe oder Breite eines Bildausschnitts.

<Double Vert> ist auch der letzte Menüpunkt des <Size>-Untermenüs.

<Flip> klappt einen definierten Bildausschnitt, je nach Wunsch über die vertikale oder horizontale Achse.

<Edge> faßt zwei Funktionen zusammen, die sich auf die Ränder eines Bildausschnitts beziehen.

Mit <Outline> wird rund um den aktuellen, definierten Bildausschnitt ein Pixelrahmen gelegt, der einen Pixel breit ist und die Farbe der aktuellen Zeichenfarbe annimmt.

<Trimm> nimmt rund um den Bildausschnitt eine Reihe Pixel weg.

<Rotate> öffnet das Untermenü, in dem es um das Drehen von Bildausschnitten geht. Die erste Option <90 Degrees> dreht den aktuellen Bildausschnitt um 90 Grad im Uhrzeigersinn.

<Any Angle> ermöglicht dem Benutzer die Drehung von Bildausschnitten in jeder gewünschten Gradzahl.

<Share> ist der letzte Punkt des <Rotate>-Untermenüs. Wenn <Share> aktiviert ist, wird die obere Kante des aktuellen Bildausschnittes fixiert, während die untere Kante verschoben werden kann. Auf diese Weise kann aus einem rechteckigen Bildausschnitt ein Parallelogramm gemacht werden.

Auch Pinsel haben bunte Seiten

Im nun folgenden Untermenü, <Change Color> sind Funktionen zusammengefaßt, die der farblichen Bearbeitung von Bildausschnitten dienen. <Bg -> Fg>. Mit dieser Option können alle Pixel eines Bildausschnittes, die in der aktuellen Hintergrundfarbe gehalten sind, in die aktuelle Vordergrundfarbe konvertiert werden.

<Bg <-> Fg> füllt alle Pixel eines Bildausschnittes, die in der Hintergrundfarbe gehalten sind, mit der aktuellen Vordergrundfarbe.

Es kann passieren, daß Sie zu einem Bild einen Bildausschnitt aus einem anderen Bild verwenden wollen. In diesem Bildausschnitt werden aber Farben verwendet, die in dieser Form nicht in Ihrem aktuellen Bild auftauchen. Wenn Sie nun <Remap> benutzen, werden alle Farben des Bildausschnittes gegen Farben aus der aktuellen Palette ersetzt. Deluxe Paint sucht dazu aus der aktuellen Palette die Farben heraus, die jenen in dem Bildausschnitt verwendeten am nächsten kommen.

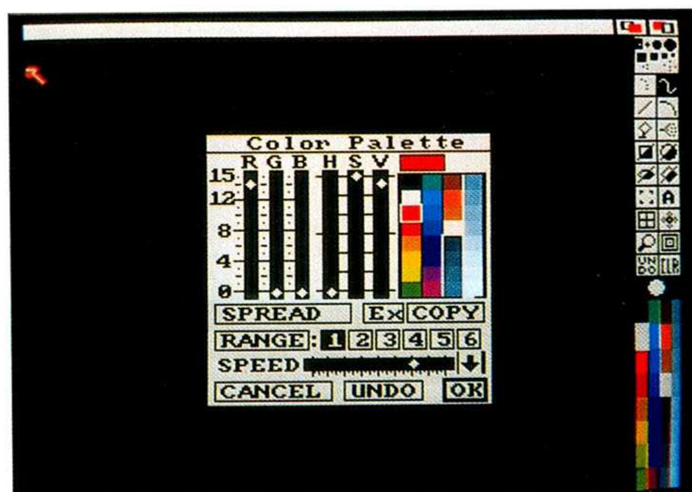
<Change Transparency> kommt dann ins Spiel, wenn Sie während der Arbeit an einem Bild eine neue Hintergrundfarbe wählen und gerade einen Bildausschnitt definiert haben, in dem die Hintergrundfarbe, die vor dem Farbwechsel aktuell war, erhalten bleiben soll. <Change Transparency> stellt auch die letzte Option des <Change Color>-Untermenüs dar.

Mit <Bend> kann ein Bildausschnitt 'gebogen' werden. Wird die Option angewählt, öffnet sich ein Untermenü, das die möglichen Richtungen, in die der Bildausschnitt gebogen werden kann, <Horiz> (Horizontal) und <Vert> (Vertikal), enthält.

Die Handhabung eines Bildausschnittes kann mitunter problematisch werden, im <Handle>-Untermenü kann angegeben werden, an welcher Seite des Bildausschnittes der Mauszeiger positioniert wird. <Center> – der Mauszeiger erscheint genau in der Mitte, über dem Bildausschnitt. Zusätzlich erscheint ein Fadenkreuz, dessen Schnittpunkt unmittelbar unter der Spitze des Mauszeigers verläuft.

<Corner> - der Mauszeiger ist in einer Ecke des Bildausschnittes positioniert. Die Ecke, an der der Mauszeiger positioniert werden soll, kann mit der Tastenkombination ALT – x bzw. ALT – y angegeben werden.

Mit <Handle> kann für den Mauszeiger eine X-beliebige



Das Herz der Farbgebung – die Palette



Ein Überblick, wie ihn Deluxe Paint nicht bieten kann. Alle Menüs und Untermenüs der Kopfleiste im Überblick

Position auf dem Bildausschnitt ausgewählt werden. Die Verfahrensweise ist einfach. Definieren Sie einen Bildausschnitt und wählen Sie <Handle> aus dem <Brush Menü>. Wenn sich das Menü wieder geschlossen hat, halten Sie die rechte Maustaste fest. Nun können Sie jede beliebige Position für den Mauszeiger auswählen.

Das Mode-Menü – Deluxe Paints mächtige Manipulatoren

Das dritte Menü in der Kopfleiste versammelt einige leistungsstarke Funktionen, wenn es um die Arbeit mit Farbe und Farbschattierungen geht.

Deluxe Paint kennt für den Umgang mit Bildausschnitten verschiedene Modi. Der erste Menüpunkt <Matte> steht für den Mode, in dem sich Deluxe Paint nach dem Einschalten befindet. Ein Bildausschnitt ist in diesem Mode ein exaktes Abbild des eigentlichen Bildes bzw. der Stelle, die ausgeschnitten wurde.

<Color> füllt den aktuellen Bildausschnitt mit der aktuellen Zeichenfarbe. Dieser Prozeß kann durch das Anwählen von <Matte> wieder rückgängig gemacht werden.

<Replc> verändert nichts am aktuellen Bildausschnitt, nur die Transparentfarbe, die normalerweise nicht sichtbar ist, wird deckend dargestellt.

Mit <Smear> können Farben auf einem Bild sozusagen verschmiert werden. Diese Funktion sollten Sie am besten ausprobieren. Zeichnen Sie zwei verschiedenfarbige Rechtecke unmittelbar nebeneinander. Wählen Sie nun einen der großen eingebauten Zeichenstifte und aktivieren Sie die <Smear>-Option. Bewegen Sie nun den Zeichenstift mit niedergedrückter linker Maustaste von einer Farbe auf die andere und beobachten Sie, was passiert.

Auch mit der <Shade>-Funktion können Farbvermischungen erzielt werden, wenn auch mit einigen anderen Vorgaben. Bei dieser Funktion stellt die aktuelle Zeichenfarbe den Mittelpunkt des Farbbereiches dar, aus dem die Farben, die gemischt werden, stammen. Wird nun die linke Maustaste betätigt, werden die Farbmischungen aus Farben erzeugt, die unterhalb des Farbbereich-Mittelpunktes liegen. Wird die rechte Maustaste benutzt, werden die darüberliegenden Farben verwendet. Diese Funktion zeigt die besten Ergebnisse, wenn mit ähnlichen Farben, z.B. mehreren Rottönen, gearbeitet wird. Ganz anders ist die Funktion von <Cycle>. Wenn dieser Mode aktiviert ist, können Sie mit allen Zeichenwerkzeugen wie bisher auch arbeiten. Nur wird nun keine einzelne Farbe mehr verwendet, sondern alle Farben der aktuellen Palette. Wenn Sie mit eingeschaltetem <Cycle>-Mode eine gerade Linie ziehen, wird diese vom Anfang bis zum Ende in allen möglichen Palettenfarben erscheinen.

Der nächste Punkt des Mode-Menüs ist <Smooth>, Deluxe Paints Weichzeichner. Die Funktion dient dazu, die Kontraste zwischen verschiedenen Farben zu verwischen, also optisch weich zu gestalten. Dabei spielt die aktuelle Bildschirmfarbe keine Rolle. Deluxe Paint sucht sich aus allen möglichen Farben die aus, die am besten passen.

Farbkontraste abgeschwächt – Deluxe Paints Weichzeichner

Die Funktion kann mit jedem eingebauten Zeichenstift, aber auch mit selbstdefinierten Bildausschnitten benutzt werden. Je nach Motiv und Palette werden die erzielten Ergebnisse unterschiedlich sein. Am besten arbeitet <Smooth>, wenn eine Farbe in mehreren Abstufungen vorhanden ist.

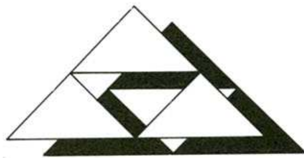
Der <Tint>-Mode kann nur mit den eingebauten Zeichen-

stiften benutzt werden. Übermalen Sie im <Tint>-Mode einen Teil eines Bildes, werden alle übermalten Pixel in den Farbton der aktuellen Zeichenfarbe konvertiert, der Helligkeitsfaktor der Farbe bleibt allerdings erhalten. Zeichnen Sie einen Farbverlauf mit mehreren Grautönen, aktivieren Sie dann den <Tint>-Mode. Wählen Sie nun einen Grünton aus der Palette, und malen Sie mit einem der großen eingebauten Pinseln auf dem vorher gezeichneten Farbverlauf herum. Sie werden sehen, daß alle grauen Pixel in grün umgewandelt werden, aber, je nach vorher aktuellem Grauton, entweder Hell- oder Dunkelgrün erscheinen. Der <HBrite>-Mode kann nur benutzt werden, wenn Deluxe Paint auf die Arbeit mit 64 Farben eingestellt ist. Auch in diesem Mode kann nicht im eigentlichen Sinne gezeichnet werden. Wird im <HBrite>-Mode mit gedrückter linker Maustaste über ein bestehendes Bild gezeichnet, werden alle betroffenen Pixel in die entsprechende Halfbright-Farbe, also dem dunklen Equivalent des Farbtones, konvertiert, sofern sie das noch nicht sind. Bei gedrückter rechter Maustaste werden die Pixel in den Farben aus den helleren 32 Farben der Halfbright-Palette verwendet.

Die Option <HBrite> ist die letzte Option des <Mode>-Menüs. Nun folgt eigentlich das <Anim>-Menü, in dem es um Animation geht. Dieses Menü werden wir allerdings nicht in diesem Kursteil behandeln, da dieses Thema zu umfangreich ist. Der vierte Teil dieses Kurses wird dann voll und ganz im Zeichen der Animation stehen. An dieser Stelle überspringen wir das <Anim>-Menü einfach und machen bei dem <Effect>-Menü weiter. Auch hier finden wir wieder einige sehr leistungsstarke und in der Praxis sehr nützliche Funktionen.

Der Farbenriegel – Mit Stencil gegen Übermalen schützen

Schon die erste Funktion <Stencil> wird für den Anwender bald zum unentbehrlichen Werkzeug. Mit <Stencil> können einzelne Farben oder ganze Farbgruppen ge-



gen das Überzeichnen mit einer anderen Farbe geschützt werden.

Wird der Menüpunkt angewählt, öffnet sich das <Stencil>-Untermenü. Die erste Option dieses Menüs ist <Make...>. Hinter dieser Option verbirgt sich das <Make Stencil>-Window.

<Make Stencil> besteht aus einer Darstellung der kompletten Palette auf der rechten Seite, und vier einstellbaren Optionen auf der linken. In der Palette klicken Sie die Farben, die verriegelt werden sollen, mit der linken Maustaste an. Daraufhin erscheint am rechten Rand der Farbe ein kleiner, schwarzer Halbkreis. Wenn Sie alle Farben, die verriegelt werden sollen, angeklickt haben, können Sie mit <Make> die Verriegelung aktivieren, außerdem verlassen Sie damit auch automatisch das <Make Stencil>-Window.

<Clear> löscht alle im <Make Stencil>-Window getätigten Verriegelungen. <Invert> invertiert den aktuellen <Stencil>, das heißt, alle bisher geschützten Farben sind nun ungeschützt, alle ungeschützten sind geschützt. Mit <Cancel> können Sie das <Make Stencil>-Window wieder verlassen, ohne etwas eingeschaltet zu haben. <Remake> kommt immer dann ins Spiel, wenn Sie eine Farbe, z.B. schwarz, verriegelt, aber danach auf Ihrem Bild noch Änderungen mit Schwarz vorgenommen haben. Es verhält sich so, daß alle Pixel, die in der Farbe schwarz gesetzt waren, als mit <Make...> ein <Stencil> definiert wurde, geschützt sind und nicht übermalt werden können. Nachträglich zugefügte schwarze Pixel sind nicht im <Stencil> enthalten und somit nicht geschützt. Nun können Sie entweder mit <Make...> erneut einen geschützten Farbbereich definieren, der nun auch alle neugesetzten schwarzen Pixel einbezieht, oder Sie können mit <Remake> einfach den momentanen <Stencil> um die neu hinzugekommenen Pixel aktualisieren.

Die <Lock FG>-Funktion läßt sich nur aktivieren, wenn der Hintergrund fixiert wurde (siehe Option <Background> weiter unten). Dann können mit <Lock FG> alle seit der Fixierung des Hintergrundes getätigten Änderungen gegen Übermalen geschützt werden, wobei die Farbe keine Rolle spielt.

Die Option <Reverse> entspricht der Funktion <Invert> im <Make Stencil>-Window. <On/Off> schaltet <Stencil>, mit dem ein Farbbereich geschützt wurde, an und aus.

Über den Menüpunkt <Free> können Sie einen <Stencil> löschen, so daß der bis dahin von ihm verbrauchte Speicherplatz wieder frei wird.

<Stencil>-Paletten können abgespeichert und wieder eingeladen werden. Die Menüpunkte <Load> und <Save> öffnen Windows, die mit den <Load>- und <Save>-Windows bei <Pics> und bei <Brush> identisch sind. Sie werden genau auf die gleiche Weise bedient. Außerdem bilden diese beiden Optionen auch den Abschluß des <Stencil> Untermenüs.

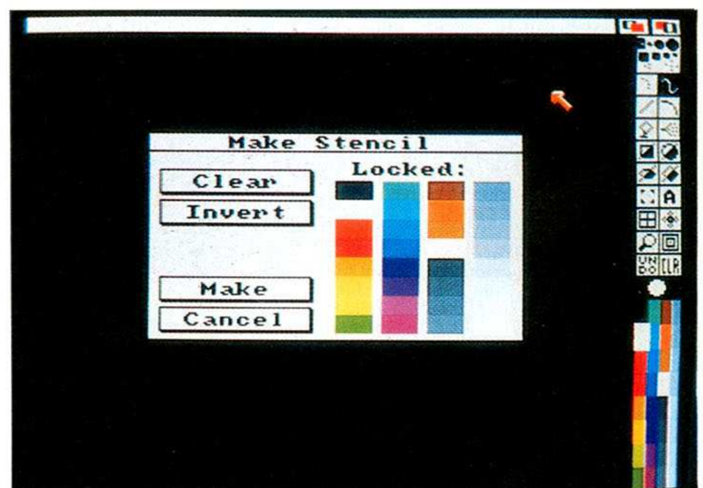
Der nächste Punkt des <Effect>-Menüs, <Background>, öffnet wieder ein kleines Untermenü, in dem die Optionen <Fix> und <Free> enthalten sind. <Fix> bewirkt, daß das gesamte, bis dahin erstellte Bild als Hintergrund fixiert wird. Das bedeutet in der Praxis, daß alle Änderungen, die nach der Fixierung zum aktuellen Bild hinzugefügt wurden, mit <CRL> gelöscht werden können, ohne daß der fixierte Bildinhalt davon betroffen wird.

<Free> annulliert die Fixierung und gibt das ganze Bild wieder frei.

Deluxe Paint in drei Dimensionen – Perspektive leichtgemacht

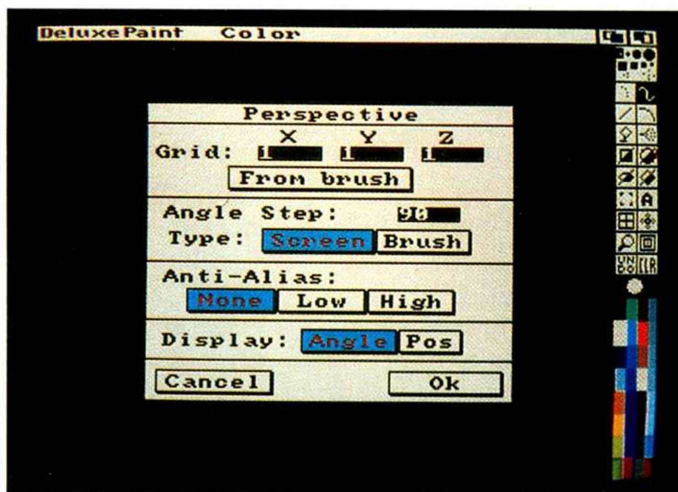
Auch der letzte Punkt des <Effect>-Menüs, <Perspective>, führt Sie wieder in ein Untermenü. Hier geht es um die perspektivischen Veränderungen an definierten Bildausschnitten. Mit <Perspective> kann ein solcher Ausschnitt über jede der drei Raumachsen gedreht werden.

Mit <Do> wird die <Perspective>-Funktion aktiviert. Wurde <Do> aktiviert, kann ein Bildausschnitt mittels des Zehnerblockes auf der Tastatur gedreht werden. Dabei werden folgende Tasten des Zehnerblocks in drei Gruppen zusammengefaßt, 7-8-9, 4-5-6 und 1-2-3. Jede Gruppe stellt eine waagrechte Tastenreihe auf dem Zehnerblock dar, und repräsentiert eine der drei Raumachsen. Wenn Sie nun z.B. die Taste 4 drücken, dreht sich Ihr Bildausschnitt nach links, drücken Sie 5, dreht er sich nach rechts. 6, die letzte Taste in der Reihe, bewirkt einen Reset, der alle vorher gegebenen perspektivischen Veränderungen, die mit den Tasten 4 und 5 verursacht wurden, ungeschehen macht. Dies gilt auch für die anderen beiden Tastengruppen. Um <Perspective> wieder zu verlassen, genügt es, wenn Sie eine normale Zeichenfunktion, z.B. <Gerade Linie>, an-



Mit Stencil können Farben gegen Übermalen geschützt werden

wählen. <Fill Screen> füllt den Bildschirm mit perspektivischen Abbildern Ihres aktuellen Bildausschnittes in den Winkeln, die vorher in <Do> eingestellt wurden.
<Reset> versetzt Ihren Bildausschnitt nach einer Bearbeitung mit <Perspective> wieder in den vorherigen Urzustand zurück.
Mit <Center> können Sie den Fluchtpunkt, auf den die perspektivische Darstellung sich bezieht, verändern bzw. neu positionieren.
Eines der wichtigsten Werkzeuge für perspektivisches Arbeiten mit Deluxe Paint verbirgt sich hinter dem letzten Punkt des <Perspective>-Untermenüs, <Settings>. Wird dieser Punkt angewählt, öffnet sich das <Perspective>-Window. Dieses Window teilt sich in vier untereinander gelegene Teile auf. Der erste ist mit <Grid> bezeichnet und zeigt drei kleine, schwarze Kästen, die mit X, Y und Z bezeichnet sind. Diese Werte repräsentieren die Parameter, mit denen Deluxe Paint ein unsichtbares, dreidimensionales Gitter erstellt. Beim ersten Anwählen dieses Windows werden diese Werte aus Ihrem aktuellen Zeichenstift übernommen.
Mit <From Brush> können die Werte eines Bildausschnittes übernommen werden.
Wie bereits erklärt, wird ein Bildausschnitt durch Tasten aus dem Zehnerblock gedreht. Wird eine der Tasten alleine gedrückt, rotiert Deluxe Paint jeweils in 1-Grad-Schritten. Unter der Option <Angle Step> kann nun ein Schrittwert eingegeben werden, der durch gleichzeitiges Drücken von Shift und der entsprechenden Taste auf dem Zehnerblock erreicht wird. Wird Shift nicht gedrückt, wird unverändert in 1-Grad-Schritten rotiert.
Ist <Screen> aktiviert, werden bei Perspective die X-, Y- und Z-Koordinaten des Bildschirms benutzt. Wenn <Brush> aktiv ist, werden die Koordinaten des Bildausschnitts verwendet.
<Anti-Alias> – Mit dieser Funktion können Sie den lästigen 'Treppchen'-Effekt vermeiden, der entstehen kann, wenn ein Bildausschnitt rotiert wird. Für diese Funktion gibt es drei verschiedene Intensitätsstufen, <None> (keine), <Low> (gering) und <High> (hoch).
Während der Arbeit mit



Perspektive im Griff, das Settings-Menü macht's möglich

<Perspective> können Sie wählen, ob in der Kopfleiste die aktuellen X-, Y- und Z-Koordinaten angezeigt werden oder ob die Position des Ausschnitts dargestellt werden soll. Unter <Display> finden Sie zwei Punkte, <Angle> und <Pos>, mit denen Sie zwischen beiden Möglichkeiten hin- und herschalten können. <Cancel> und <OK> bedürfen wohl keiner gesonderten Erklärung.

Das Prefs-Menü – Werkzeuge zum Ein- und Ausschalten

Mit <Prefs> erreichen wir den letzten Menüpunkt der Kopfleiste. Hier finden sich einige Zeichenwerkzeuge, die nach Belieben ein- bzw. ausgeschaltet werden können. Wenn in diesem Menü eine Funktion aktiv ist, erscheint ein Stern vor dem betreffenden Menüpunkt.
<Coords> schaltet eine Koordinatenanzeige in der Kopfleiste ein bzw. aus.
Wer mit großen Bildausschnitten arbeitet, wird recht bald feststellen, daß die Arbeit damit recht problematisch werden kann. <Fast FB> ersetzt bei Zeichenaktionen mit <Gerader Linie> oder anderen 'Linien'-Werkzeugen den Bildausschnitt gegen eine einfache Linie, gerade so, als würden Sie mit dem kleinsten eingebauten Zeichenstift malen. Im fertigen Resultat ist aber dann der Bildausschnitt zu sehen. Mit dieser Funktion können einige lästige Wartezeiten und zähes Arbeiten vermieden werden.
<MultiCycle> hat einiges mit der <Cycle>-Funktion aus dem <Mode>-Menü gemeinsam. Aber während <Cycle> nur mit einem zu einer Fläche umgestalteten Bildausschnitt benutzt werden kann, wirkt <MultiCycle> auf den Bildausschnitt in seinem ursprünglichen Aussehen. Dabei werden alle in diesem Ausschnitt verwendeten Farben 'durchgerollt'. Am besten definieren Sie einen Bildausschnitt, aktivieren <MultiCycle> und experimentieren damit ein wenig herum.
<Be Square> – Da die Pixeldarstellung des Amigas nicht exakt quadratisch ist, kann es beim Zeichnen von Kreisen und Quadraten zu optischen Verzerrungen kommen. Wenn Sie dies ausschalten wollen, aktivieren Sie <Be Square>, dann berücksichtigt Deluxe Paint die Pixelungenauigkeiten. Mit der Funktion <Workbench> können Sie zwischen Deluxe Paint und der Workbench hin- und herschalten, z.B. um eine Diskette zu formatieren.
Die Funktion <ExclBrush> kann nur mit eingeschaltetem <Gitter> benutzt werden. Sind diese beiden Punkte aktiv und ein Bildausschnitt wird definiert, läßt Deluxe Paint an dessen unterer und rechter Seite eine Reihe Pixel verschwinden. Die Transparentfarbe ist nicht leicht zu handhaben, aber auch hier haben die Deluxe-Paint-Macher an Werkzeuge ge-

Haben Sie die richtige Lösung gefunden?

Sicherlich ist es Ihnen gelungen, zu allen Aufgaben, die wir in der letzten Ausgabe gestellt hatten, die richtige Lösung zu finden.

Frage 1: Change-Color-Menü aufrufen und Palette wählen. Im Palette-Menü die erste Farbe des gewünschten Farbbereiches und danach Range anklicken. Die Palette mit OK wieder schließen. Das 'Füllen'-Icon mit der rechten Maustaste anklicken. Im Fill Type Requester hinter Gradient das letzte Kästchen aktivieren. Den Fill Type Requester wieder schließen und eine Farbe anwählen, die im definierten Farbbereich liegt. Gefüllten Kreis zeichnen.

Frage 2: Das Icon 'Gefülltes Vieleck' aktivieren, Farbe wählen und Form zeichnen. Anfangs- und Endpunkt dieser Form müssen identisch sein.

Frage 3: Icon 'Zeichenstift definieren' aktivieren. Gewünschten Bildausschnitt als Zeichenstift übernehmen. Das 'Fill'-Icon einmal mit der rechten Maustaste anklicken. Im Fill Type Requester, der sich daraufhin öffnet, die Option 'From Brush' anklicken. Fill Type Requester schließen und Füll-Funktion benutzen.

Frage 4: Das 'Gerade Linie'-Icon mit der rechten Maustaste einmal anklicken. Im Spacing Requester Airbrush aktivieren und dahinter '5' eingeben. Requester schließen und gerade Linie zeichnen. Den Vorgang wiederholen, nur diesmal im Zahlenfeld 25 eingeben.

Frage 5: Das 'Gitter'-Icon einmal mit der rechten Maustaste anklicken. Im Gridding Requester X- und Y-Spacing auf 10 einstellen. OK anklicken. Gerade Linie anwählen, Gitter aktivieren und Matrix zeichnen. Überflüssige Pixel löschen.

dacht. <AutoTransp> hilft Ihnen, auch eine Fläche, die mit der Transparentfarbe gefüllt ist, als Bildausschnitt zu benutzen. Wenn <AutoTransp> aktiv ist, checkt Deluxe Paint jedesmal, wenn Sie einen Bildausschnitt entnehmen, ob alle vier Eckpunkte in derselben Farbe gesetzt sind. Ist dies der Fall, wird diese Farbe automatisch in die Transparentfarbe umgewandelt.

<No Icons> – Üblicherweise speichert Deluxe Paint Bilder und Bildausschnitte mit einem Icon ab, so daß Sie auch von der Workbench aus nachsehen können, welche Bilder auf einer Diskette abgespeichert sind.

<No Icons> schaltet dies ab, so daß Bilder und Bildausschnitte ohne Info-Datei, also ohne Icon, abgespeichert werden.

Das waren die Funktionen der Kopfleiste, so wie sie der Anwender unmittelbar nach dem Start von Deluxe Paint sieht. Natürlich konnten wir in diesem Teil unseres Deluxe-Paint-Kurses wenig mehr tun, als auf jede hier enthaltene Funktion einen kurzen Blick zu werfen. Aber es gibt mehr als eine Funktion, die einer genaueren Erklärung bedarf. Der nächste Teil dieses Kurses wird sich noch einmal ausgiebig mit der Kopfleiste von Deluxe Paint befassen.

Wie sicher sind Sie?

Auch diesmal wollen wir Ihnen mit diesen Fragen bei der Überprüfung Ihres Wissens helfen.

Leider waren in diesem zweiten Teil so viele Funktionen Thema, daß es, um diese alle abzufragen, eines extra Heftes bedürfte. Sie sollten also auf eigene Faust versuchen, die eben erläuterten Funktionen noch weiter zu erkunden und ein wenig mit ihnen zu experimentieren. Dann sollten Ihnen auch die Antworten auf die folgenden Aufgaben nicht allzu schwerfallen.

1.) Definieren Sie einen Bildausschnitt und wenden mit diesem die <Perspective>-Option <Fill Screen> an.

2.) Zeichnen Sie zwei verschiedenfarbige Rechtecke, und verschmieren Sie die Farben mit der <Smear>-Option.

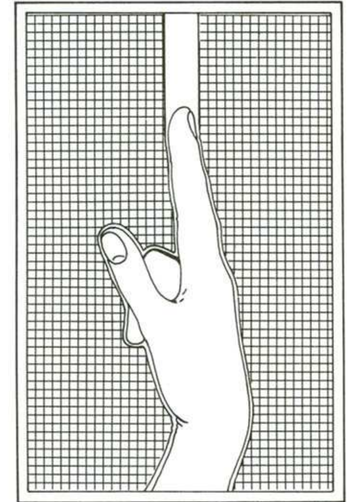
3.) Erstellen Sie ein Schachbrettmuster, und sichern Sie die weiße Farbe mit <Stencil> gegen Übermalen.

4.) Erstellen Sie ein beliebiges Bild, und kopieren Sie es in den <Spare>-Bildschirm.

5.) Löschen Sie Ihren aktuellen Zeichenbildschirm, schalten Sie in den <Spare>-Bildschirm, und definieren Sie dort einen Bildausschnitt. Übernehmen Sie den im <Spare>-Bildschirm definierten Bildausschnitt in den leeren Bildschirm.

Wir wünschen Ihnen viel Spaß beim Ausprobieren des eben Gelernten und hoffen, Sie auch im nächsten Heft wieder begrüßen zu dürfen.

(hs)



2400 Baud MODEMS



TORNADO 2400 E
Tischgerät incl. Steckernetzteil für alle Rechner mit RS 232/V.24 **399,–**

MAXMODEM 2400 MNP 5
Wie oben, mit MNP5 Übertragungsprotokoll bis 4800 bps eff. Geschw. **599,–**

PC-Karte, halbe Länge, COM 1: bis COM 4: konfigurierbar **349,–**

Lieferung per Nachnahme, komplett mit engl. Handbuch und Telefonkabel (USA). 1 Jahr Garantie, Rückgaberecht innerh. von 8 Tagen ohne Angabe von Gründen.

Leistungsmerkmale:
2400, 1200, 600, 300 Baud CCITT V.22 bis V.22/V.21, 1200 und 300 Baud Bell 212a/103, kompatibel mit HAYES SMARTMODEM 2400 (AT-Kommandos), automatische Wahl (Ton oder Impuls), Autoanswer, Konfiguration speicherbar, basierend auf INTEL Chipsatz.
Der Betrieb dieser Modems am öffentlichen Postnetz in der BRD und in West-Berlin ist verboten und unter Strafe gestellt.

Carl Schewe (GmbH & Co.), Abt. Modems
Essener Str. 97, 2000 Hamburg 62
Tel. (0 40) 5 27 03 21, Fax (0 40) 5 27 66 54

WIR HABEN COMPUTER, ZUBEHÖR, LITERATUR...
ZUM SPIELEN, LERNEN, ARBEITEN...



BESUCHEN SIE UNS ODER FORDERN SIE KOSTENLOSE PREISLISTE AN!
B&C EDV-Systeme GesmbH
Tel.: 0222/5054978 Fax: 0222/5054978
Mo-Fr 9⁰⁰-12⁰⁰ u. 13⁰⁰-18⁰⁰ Sa 9⁰⁰-12⁰⁰

Schnell... mit integriertem RGB-Splitter ... nur 698,- DM!
der ganz andere Video-Digitizer
Digi Tiger
by KLAUS D. TUTE
Soft-Art- und Hardware
Mathildenstr.12 3000 Hannover 91
05 11 / 62 98 25
Diskette mit Demobildern und Demoprogramm gegen 10 DM Schein

Nachdem Markt& Technik mit dem Buch "Programmieren in AmigaBASIC" Anfängern auf die Sprünge half, sind nun die Profis an der Reihe. Der Titel läßt auf knallharte Informationen schließen, doch der Schein trügt. Dieses Buch ist offenbar nicht für Profis gedacht, sondern eher für diejenigen, die es erst werden wollen. Der Profi würde sich bei den meist überflüssigen und unpassenden Formulierungen bestimmt langweilen. Selbst dem Einsteiger können die oft vorkommenden Füllsätze und Anhängsel irgendwann lästig werden beziehungsweise letztendlich sogar völlig verwirren. Wen "schwammige Formulierungen" jedoch nicht stören, dem kann man dieses Buch empfehlen.

AmigaBASIC für Profis

Das Buch "AmigaBASIC für Profis" beschäftigt sich zum einen mit diversen BASIC-Problemen, geht aber größtenteils auf die Programmierung der Hardware und der Benutzeroberfläche "Intuition" ein. Dabei wird das Thema "Library" großgeschrieben. Bereits nach 23 Seiten wird dem Leser das Wissenswerte über diese nützliche Einrichtung vermittelt. Danach werden die Intuition- und die Graphics-Library besprochen. Die DOS-Library folgt vier Kapitel später.

Der Profi, der sich in diesem Gebiet vermutlich ebensogut auskennt wie in Assembler, wird hier mit der sogenannten "harten" Lektüre besser bedient sein. Dem fortgeschrittenen BASIC-Programmierer liefern diese Kapitel allerdings einen guten Einblick in die Betriebssystemroutinen und deren Verwendung vom BASIC aus. Des weiteren geht dieses Werk auf die Themen Zeichensätze, Copper und Animation ein, womit eigentlich der Bereich Grafik im großen und ganzen abgedeckt ist.

Es verbleibt nur noch das Kapitel Sound, das leider nur sehr kurz angesprochen wird. Es geht dabei hauptsächlich um die Theorie und das Abspielen digitalisierter Sounds. Abschließend erläutern die Autoren, was es mit dem Stichwort 'Künstliche Intelligenz' auf sich hat. Insgesamt hinterläßt das Buch einen recht positiven Eindruck, wenn man von den anfangs erwähnten Schwächen einmal absieht. Den Titel "AmigaBASIC für Profis" verdient dieses Werk allerdings nicht.

Der Profi sollte sich besser einer Lektüre widmen, die die Informationen wesentlich kompakter übermittelt.



Dem Fortgeschrittenen, der sich weiterbilden möchte, kann dieses Buch jedoch bedenkenlos empfohlen werden, da es ihm gute Ideen und Informationen liefern kann. Durchaus positiv zu werten ist, daß alle besprochenen Programme auf zwei beiliegenden Disketten nach Kapiteln geordnet vorhanden sind. Das spart Zeit und Nerven.

(Bernd Rudolf/br)

Name: AmigaBASIC für Profis
Autoren: Amir R. Amir und Martin Höfler
Verlag: Markt&Technik
ISBN-Nr.: 3-8909-071-05
Preis: 79,– DM

Assembler hat für viele Programmierer einer Computersprache etwas Mystisches, ist sie doch die einzige Sprache, die der Prozessor nur verstehen kann. Logischerweise arbeitet dadurch der Computer Maschinensprachprogramme auch wesentlich schneller ab als Programme, die erst übersetzt werden müssen.

Nur wenige Programmierer steigen jedoch von ihrer einmal erlernten Computersprache auf Assembler um, da sie den Komfort und die Bequemlichkeit "ihrer" Sprache nicht missen möchten. Denn in Assembler zu programmieren heißt, elementar zu programmieren, also jedes Byte einzeln zu setzen.

Maschinensprache Amiga

Assembler ist aber auch für die "Hochsprachler" interessant, da viele Sprachen die Möglichkeit bieten, Maschinensprachroutinen in den Source-Code zu implementieren. Auf diese Weise kann z.B. ein langsames Interpreterprogramm enorm an Geschwindigkeit gewinnen, wenn man nur einige zeitrelevante Unterprogramme durch entsprechende Aufrufe von Assemblerroutinen ersetzt. Um den Einstieg in diese enorm schnelle und direkte Sprache zu erleichtern, hat Data Becker das Buch Maschinensprache Amiga herausgegeben.

Der Autor beschreibt in einem aufgelockerten Stil zunächst einmal den Aufbau des Amiga, die Speicherstruktur und die Unterschiede zwischen den verschiedenen Speicherbausteinen wie RAM, ROM und WOM (beim Amiga 1000).

Das nächste Thema sind die Unterschiede zwischen Bits, Bytes, Words und Longwords und die für ein Maschinensprachebuch schon obligatorische Einführung in die verschiedenen Zahlensysteme.

Der große Bereich des Buches klärt Grundlegendes zum Hauptprozessor MC68000. Dabei wird unter anderem auf die Adressierungsarten und den Befehlssatz genauer eingegangen. Anschließend erfolgt eine Einweisung in die Bedienung der Assembler Profimat, SEKA und "Assem". Das Durcharbeiten dieses Kapitels erspart so manchen Blick ins Handbuch, da alle Besonderheiten der bereits erwähnten Assembler klar beschrieben werden. Die abgedruckten Programme beziehen sich übrigens auf den SEKA-Assembler.

Nach dem Vorgeplänkel geht es ran an den Speck: Die ersten Programme warten darauf, abgetippt zu werden.

Um Ausgaben auf den Bildschirm projizieren zu können, muß zuerst ein Screen bzw. ein Window geöffnet werden. Um dies einfacher zu gestalten, beinhalten die Programme Bytemanipulationen, deren Ergebnisse man nach Ablauf des Programms aus den Registern des Prozessors auslesen kann. Weitere Programme zum Abtippen können beispielsweise Werte aus den verschiedenen Zahlensystemen (BIN-DEZ, HEX-BIN) umrechnen, Tabellen sortieren, Werte addieren und ähnliches.

Nachdem auf diese Weise eine erste Einführung in die Sprache und deren Befehle

geschehen ist, wendet sich der Autor den Hardwareregistern des Amiga zu. Hier wird erklärt, wie man zum Beispiel Maus- und Joystickport auslesen oder die laufende Zeit ermitteln kann. Auch erfährt der Leser Grundlegendes zur Tonerzeugung.

Die nächsten beiden Kapitel wenden sich endlich dem Betriebssystem des Amiga und der Arbeit mit Intuition zu.

Des weiteren folgen Beispiele zum Umschalten in den Supervisormodus, zur Exeptions- und rekursiven Programmierung. Im Anhang sind noch einmal alle Funktionen der Libraries und die Befehle des Prozessors aufgelistet.

Das Buch ermöglicht einen recht einfachen Einstieg in Assembler, ist jedoch etwas zu sprunghaft aufgebaut. Zum besseren Verständnis muß man das Buch mindestens zweimal lesen, da an vielen Stellen im Buch auf bestimmte Kapitel vorgegriffen wird, was den Lesefluß (und damit auch das Verständnis) ziemlich beeinträchtigt. Weiterhin fällt schon nach wenigen Seiten auf, daß oftmals auf weiterführende Literatur zurückgegriffen werden muß, da nicht genauer auf wichtige Themen wie zum Beispiel die Exeptionsprogrammierung eingegangen wird oder aber Parameter von Funktionen nicht richtig erläutert werden. So ist das nachfolgende Zitat in abgewandelter Form leider des öfteren im Text anzutreffen:

"Das nun folgende Wort ist wieder so aufgebaut, daß manches Bit eine eigene Bedeutung hat. Mit diesem Wort wird gewählt, welcher Art der Screen sein soll. Setzen wir hier eine 15 ein, so wird unser Screen ein "Customscreen", was uns alle Möglichkeiten bietet."

Mit diesen Angaben wird der Leser leider alleingelassen. Welche Ergebnisse andere Werte bringen, muß man gegebenenfalls selbst ausprobieren oder in weiterführender Literatur nachlesen.

(R.Marz/br)

Name: Maschinensprache Amiga
Autor: Stefan Dittrich
Verlag: Data Becker
ISBN: 3-89011-076-2
Preis: 49,– DM

Das vorliegende Buch wendet sich, wie der Titel schon besagt, an den Amiga-Einsteiger und beschreibt in betont lockerer Form all das, was dieser wissen sollte und in der spärlichen Dokumentation des Amiga-Handbuchs nicht findet.

Das erste Kapitel beschreibt den Inhalt der Verpackung des Amiga 500: Der korrekte Anschluß der Hardware wird für den Amiga-Anfänger deutlich gemacht und die Verkabelung ausführlich erklärt.

Amiga 500 für Einsteiger

Nachdem die Hardware installiert wurde, geht es im nächsten Kapitel um Grundsätzliches. Hier wird dem Einsteiger die typische Amiga-Umgebung Intuition erklärt. Er lernt, mit der Maussteuerung zu arbeiten, das richtige Umgehen mit Fenstern, kurz all jenes, was der Amiganer braucht, um mit dem Amiga-System klarzukommen (insbesondere der Umgang mit den Preferences).

Das zweite Kapitel wendet sich ausschließlich dem AmigaBASIC zu. Hier werden dem Leser Grundlagen über die Arbeit mit dem AmigaBASIC vermittelt. Das Kapitel enthält zwar Hinweise zum Umgang mit dieser Computersprache, sowie einige kleine Beispielprogramme, kann (und soll) das Handbuch jedoch nicht ersetzen.

Der nächste Abschnitt beschäftigt sich mit dem Betriebssystem AmigaDOS. Hier werden der Command Line Interpreter (CLI) besprochen und einige Tips dazu gegeben. Leider ist dieser Abschnitt einer der dünnsten im ganzen Buch. Gerade der Einsteiger hat mit der Bedienung des CLI seine Schwierigkeiten, so daß dieser Teil des Buches größere Beachtung von seiten des Autors verdient hätte.

Das letzte Kapitel behandelt auftretende Fehler, erläutert diese und hilft bei deren Beseitigung. Ein kleines Wörterbuch bringt dem Einsteiger die einschlägigen Fachbegriffe näher. Den Abschluß bilden Informationen zum Thema Btx und Amiga.

Das Buch kann ohne weiteres allen Einsteigern uneingeschränkt empfohlen werden, wenn man von der kleinen Schwäche, das Thema CLI betreffend, einmal absieht.

(Klaus Rupp/br)

Name: Amiga 500 für Einsteiger
Autor: Christian Spanik
Verlag: Data Becker
ISBN 3-89011-192-0
Preis: 39,– DM

Das Buch beschäftigt sich mit der Hardware der Amiga-Systeme und deren Programmierung. Hierbei ist zu beachten, daß die Autoren auf die drei Amiga-Modelle (500, 1000, 2000) eingehen.

Gleich zu Anfang bekommt der Leser eine Einführung in die Bedienung des Seka-Assemblers und der beiden gängigsten C-Compiler Lattice und Aztec. Ein kleines Batchfile erleichtert dem Neuling dabei die Arbeit mit den Compilern ungemein.

Amiga System-Handbuch

Im Anschluß daran wird kurz auf die verschiedenen Ausführungen der Amiga-Reihe eingegangen und die Unterschiede anhand von Blockdiagrammen verdeutlicht. Zwei weitere Kapitel behandeln den Bootvorgang und die CPU MC68000, bevor dann in einem speziellen Kapitel auf die drei Custom-Chips Agnus bzw. Fat Agnus, Paula und Denise eingegangen wird.

Hier erfährt der Leser das Wichtigste über Pinbelegungen und die Funktionsweise dieser Spezialchips. All das Wissen wird durch Blockschaltbilder und kleine Beispielprogramme, die auf der beiliegenden Diskette enthalten sind, unterstützt.

Selbstverständlich werden auch Blitter und Copper ausführlich in ihren Funktionen beschrieben und durch die Programme auf der Diskette verdeutlicht.

Ein weiteres Kapitel beschäftigt sich mit den Steckplätzen der verschiedenen Amiga-Modelle. Dargestellt werden die 86-Pin-Slots aller Amigas. Der Schwerpunkt liegt, wie sollte es auch anders sein, bei den 100-Pin-Slots des B-Amiga-2000. Die Slots werden ausführlich dargestellt und deren Pinbelegung in Tabellen verglichen. Im weiteren wird auch auf die Belegungen der PC- bzw. der AT-Slots eingegangen.

Der nächste Abschnitt widmet sich der Floppy, wobei unter anderem auf die verschiedenen Aufzeichnungsverfahren eingegangen wird.

Das Kapitel Schnittstellen ist für diejenigen wichtig, die ihre Peripherie-Bausteine bei einem Modellwechsel weiterbenutzen möchten. Dabei werden die Unterschiede in den Belegungen der verschiedenen Amiga-Modelle herausgearbeitet.

Ein weiterer Abschnitt (meiner Meinung nach ein kleiner Leckerbissen) ist Bastelanleitungen gewidmet. So ist es möglich, für relativ wenig Geld einen Digitizer, einen Scanner, ein Schaltinterface und sogar ein kleines Genlock zu basteln. MS-DOS-Fähigkeiten beschreibt das nächste Thema: Hier werden das Sidecar, die XT- und AT-Karte sowie deren Speicher- und I/O-Belegung ausführlich erläutert. Das Kapitel über RAM-Erweiterungen behandelt die verschiedenen Speichererweiterungen und die Unterschiede zwischen dynamischem und statischem RAM. Last but not least wird die Expansion-Library behandelt, die nötig ist, um Hardware-Erweiterungen korrekt in das System einzubinden.

Für den, der sich intensiv mit der Hardware des Amiga und dessen Betriebssystem beschäftigen möchte, ist dieses Buch ein Muß. Die Bastelanleitungen tragen ihren Teil dazu bei, den Inhalt interessant zu machen. Nicht verschweigen möchte ich aber auch den hervorragenden Bildteil in der Mitte des Buches. Hier findet der Leser Farbfotos aller Hardwareteile, die im Buch vorgestellt werden, was vor allem dem Einsteiger die Orientierung auf der Platine erleichtert.

(Klaus Rupp/br)

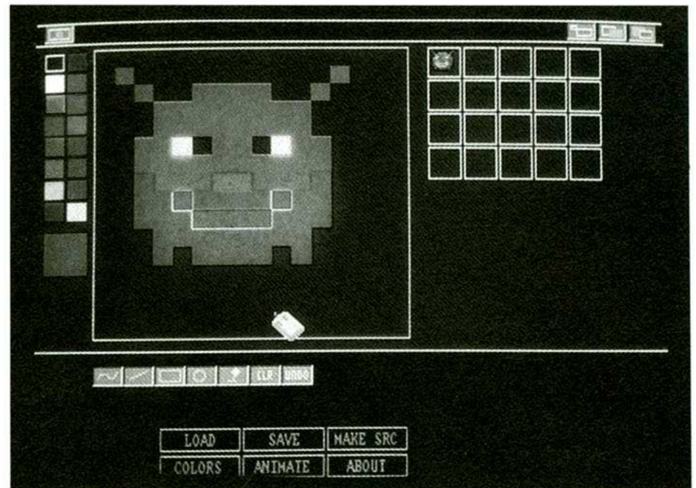
Name: Amiga System-Handbuch
Autor: Kremser, Koch
Verlag: Markt&Technik
ISBN: 3-89090-550-1
Preis: 79,– DM

Garry Glendown

SprEd Das Sprite-Editor-Projekt

Teil 1: Das Grundgerüst

Wenn man auf dem Amiga Sprites erstellen will, hat man für ihre Erstellung keine angemessenen Tools zur Auswahl. Der einzige gute Weg führt über ein Malprogramm, und danach über ein Utility, das die IFF-Files in den Source für eine Programmiersprache umwandelt. Das klingt natürlich nicht nur nach einem Umweg, sondern ist auch einer...



Sprites, Bobs und Images sind das A und O bei Spielen. Mit SprEd steht Ihnen ein Tool zur Verfügung, das die Ergänzung dieser Objekte vereinfacht

Dem soll unser SprEd-Projekt Abhilfe leisten. Es handelt sich dabei um ein Programm, mit dem man Sprites, Bobs und Images erstellen und als Source für beliebige Sprachen ausgeben kann. Im ersten Teil wird das Grundgerüst erstellt, das alle Definitionen für die Benutzeroberfläche sowie die ersten Fragmente für deren Abfrage enthält. Außerdem wird bereits der File-Requester aufgerufen, der von der Arp-Library zur Verfügung gestellt wird. Diese Arp-Library befindet sich unter anderem auf den Public-Domain-Disketten Fish 123 und Panorama 27c und muß auf jeden Fall verfügbar sein.

Ein Editor mit 16 Farben – die wahre Pixelpracht

Doch nun zum Programm. Da wir nicht nur 4farbige Sprites erstellen wollen, muß ein Screen mit 16 Farben geöffnet werden. Dies wird mit der NewScreen-Struktur am Anfang des zweiten Listings vorbereitet (siehe Box 1).

Die ersten vier Werte enthalten die Größe des Screens; die '4' definiert die Anzahl der Bitplanes (2^4 = 16 Farben). Am Ende des Listings findet man die zweite wichtige Struktur, 'NewWindow'. Hier werden verschiedene Parameter für die Benutzerobefläche vorbereitet (siehe Box 2).

Mit GADGETUP und GADGETDOWN sagen wir, daß das Fenster eine Nachricht schicken soll, sobald ein Gadget gedrückt und wieder losgelassen wird. Mit den anderen Flags in der nächsten Zeile wird der Rahmen abgeschaltet, das Fenster wird automatisch aktiviert, und beim Drücken der rechten Maustaste erscheint kein Balken mehr am oberen Bildrand.

'CUSTOMSCREEN' am Ende der Struktur sorgt dafür, daß das Fenster nicht auf der Workbench, sondern der eigene Screen geöffnet wird.

Die gesamte Benutzeroberfläche wird in diesem Programm über Gadgets geregelt, daher an dieser Stelle eine etwas genauere Erklärung der Struktur 'Gadget' (siehe Box 3).

In NextGadget befindet sich ein Zeiger auf das nächste Gadget. Die nächsten vier Werte definieren die Position und Größe des Gadgets. In den Flags werden verschiedene 'Erscheinungsformen' des Gadgets eingestellt. In unserem Programm werden dabei folgende verwendet:

GADGHBOX – Wird das Gadget selektiert, so wird ein Rahmen darum gezeichnet.

GADGHIMAGE – Wie GADGHBOX, nur wird hier ein eigenes Image angezeigt. Achtung! Falls diese beiden zusammen gesetzt werden, wird das Gadget beim Selektieren nicht verändert.

GADGIMAGE – Statt eines Textes oder eines Rahmens wird eine Grafik dargestellt.

Im Feld 'Activation' werden Zusatzinformation für das Gadget gesetzt. Bei SprEd wird dabei nur das Flag 'RELVERIFY' verwendet, das dazu dient, nur dann eine GADGETUP-Message zu schicken, falls der Mauszeiger beim Loslassen noch über dem Gadget war. Damit wird ermöglicht, daß der Benutzer das Anklicken eines Gadgets 'rückgängig' machen kann, indem er einfach die Maus bewegt und dann den linken Knopf losläßt.

Bei dem Gadget, in dem das Zeichnen der Grafik stattfindet, werden außerdem noch die Flags GADGIMMEDIATE und FOLLLOWMOUSE gesetzt. Damit wird erreicht, daß wir jede Bewegung der Maus erhalten und damit das Setzen der Punkte erleichtert wird (mehr dazu im nächsten Teil).

In GadgetType wird die Art des Gadgets eingestellt. Wir haben dabei nur BoolGadgets benutzt, also 'Knöpfe', die entweder gedrückt oder nicht gedrückt sind.

In GadgetRender und SelectRender werden entweder Zeiger auf Border-Strukturen oder Image-Strukturen eingetragen. Ersteres dient für die 'normale', unselektierte Darstellung, letzteres wird beim Selektieren des Gadgets dargestellt.

GadgetText enthält einen Zeiger auf eine IntuiText-Struktur. Hiermit kann man beliebige Texte für jedes Gadget ausgeben.

MutualExclude ist z.Zt. noch ohne Bedeutung, deshalb sollte man hier 'NULL' eintragen.

SpecialInfo kommt nur zum Einsatz, falls Proportional- oder StringGadgets verwendet werden.

```
struct NewScreen NewScreenStructure = {
     0,0,
     640,256,
     4,
     0,1,
     HIRES,
     CUSTOMSCREEN,
     NULL,
     NULL,
     NULL,
     NULL};
```

Box 1

```
struct NewWindow NewWindowStructure1 = {
 0,0,
     640,256,
     0,1,
     GADGETDOWN|GADGETUP,
BORDERLESS|ACTIVATE|RMBTRAP|NOCAREREFRESH,
&Gadget1,
     NULL,NULL,NULL,NULL,
     5,5,-1,-1,
CUSTOMSCREEN};
```

Box 2

Die letzten beiden Felder enthalten Werte, die von Intuition ignoriert werden und so Informationen für den Programmierer enthalten können.

Soweit zu dem File 'SprEd.h'. In dem File 'Defines.h' werden einige Labels für die Identifikation der Gadgets definiert. Doch nun zum Hauptprogramm.

Nun geht's rund: das Hauptprogramm

Im Hauptprogramm wird zuerst die Routine 'OpenStuff' aufgerufen. Diese holt alle Libraries (Intuition, Graphics, Arp). Danach wird der Screen geöffnet. Damit das Window auch dort erscheint, wird der Screen-Pointer in die Window-Struktur eingetragen. Nachdem auch das Fenster aufgemacht ist, kümmern wir uns noch um einen kleinen 'Leckerbissen': eine Copper-Liste, die die verschiedenen Farben einstellt. Sobald alle Copper-Befehle eingetragen sind, wird mit 'RethinkDisplay' die neue Copper-Liste eingebunden.

In der Hauptroutine kommt eine Quasi-Endlosschleife, in der die Nachrichten vom Fenster abgefragt werden. Wichtig ist dabei die Zeile 'Waitport(win->UserPort)'. Damit wird das sogenannte 'Busy-Wait' verhindert, so daß unser SprEd beim Warten keine Rechenzeit verbraucht.

Danach folgt der Block, in dem die ganzen Intuition-Messages abgearbeitet werden. Dort folgen zunächst die Abfragen für die Maustasten. Um Punkte zu setzen, wird im Hauptprogramm das Flag 'DRAWIT' verwendet. Dieses wird beim Loslassen der linken Maustaste gelöscht. Ähnlich werden auch die beiden möglichen Aktionen der rechten Maustaste verarbeitet, jedoch wird hier noch ein weiteres Flag verwendet, nämlich 'ERASE'. Ist dieses Flag gesetzt, so werden die Punkte in der Farbe '0' gemalt, also gelöscht.

Wie man bei 'MENUDOWN' noch sieht, werden die beiden Routinen 'CopyToUndo' und 'Pressed' aufgerufen. Die erste Funktion kopiert das aktuelle Bild in den Undo-Buffer, die zweite sorgt dafür, daß der Punkt, an dem die rechte Maustaste gedrückt wurde, gelöscht wird.

Als nächstes folgt der große 'GADGETUP'-Block, also der Code für die Verarbeitung von losgelassenen Gadgets. Die ersten vier Case-Blöcke bearbeiten dabei die Gadgets zum Schließen, Screen nach hinten und vorne sowie die 'DragBar'.

Wurde das CloseGadget angewählt, so wird die Schleife um die Message-Abfrage verlassen, und die Routine 'ex' zum Beenden wird angesprungen. Dort werden alle Libraries geschlossen sowie Screen, Window und der Speicher für die Copper-Liste freigegeben.

Wurde eines der Farb-Gadgets angewählt, wird die neue Mal-Farbe aus dem Feld 'UserData' der Gadget-Struktur geholt. Danach wird das Anzeigefeld für die aktuelle Farbe neu gezeichnet. Dabei wird allerdings noch zwischen der Farbe '0' und den anderen differenziert: Ist die neue Farbe '0', so wird zuerst ein Rahmen in der Farbe '1' um das Feld gezeichnet.

Die letzte Abfrage im 'GADGETUP'-BLock kümmert sich um die einzelnen Images. Auch hier wird, wie bei den Farben, die Nummer des Gadgets aus dem UserData-Feld geholt. Außerdem wird der Offset zum Malen in den Images neu bestimmt.

Soweit der heutige Teil. Im nächsten Heft widmen wir uns dann den Routinen für die anderen Mal-Funktionen. Bis dahin viel Spaß beim Abtippen.

(br)

Wichtig!

Das Programm SprEd benötigt auf jeden Fall die Arp-Library. Diese ist Public Domain und darf deswegen nicht abgedruckt bzw. auf der Diskette zum Heft abgespeichert sein. Wenn Sie SprEd lauffähig machen wollen, besorgen Sie sich bitte zuerst die Arp-Library. Sie befindet sich unter anderem auf der Fishdisk 123 und auf der Panorama 27c.

```
struct Gadget
{
      struct Gadget *NextGadget;
      SHORT LeftEdge, TopEdge;
      SHORT Width, Height;
      USHORT Flags;
      USHORT Activation;
      USHORT GadgetType;
      APTR GadgetRender;
      APTR SelectRender;
      struct IntuiText *GadgetText;
      LONG MutualExclude;
      APTR SpecialInfo;
      USHORT GadgetID;
      APTR UserData;
}
```

Box 3

Bitte die Zeilennummern nicht mit abtippen, sie dienen lediglich der Übersicht.

Listings

```
****************************************************************
*                        AMIGA DOS INFO                        *
*   Programmname        : SprEd.h                              *
*   Funktion            : siehe Text                           *
*   Sprache             : Aztec-C                              *
*   Besonderheiten      : keine                                *
****************************************************************

struct TextAttr TOPAZ80 = {
 (STRPTR)"topaz.font",
 TOPAZ_EIGHTY,0,0};
struct NewScreen NewScreenStructure = {
 0,0,640,256,4,0,1,HIRES,CUSTOMSCREEN,NULL,NULL,NULL,
 NULL};
#define NEWSCREENSTRUCTURE NewScreenStructure
SHORT BorderVectorsb1[] = {0,0,80,0,80,12,0,12,0,0};
struct Border Borderb1 = {-1,-1,3,0,JAM1,5,
 BorderVectorsb1,NULL};
struct IntuiText ITextb1 = {1,0,JAM2,6,2,&TOPAZ80,
 (UBYTE *)"MAKE SRC",NULL};
struct Gadget Gadgetb56 = {NULL,293,228,79,11,GADGHBOX
 ,
 RELVERIFY,BOOLGADGET,(APTR)&Borderb1,NULL,&ITextb1,
 NULL,NULL,MAKESRC,NULL};
SHORT BorderVectorsa1[] = {0,0,80,0,80,12,0,12,0,0};
struct Border Bordera1 = {-1,-1,3,0,JAM1,5,
 BorderVectorsa1,NULL};
struct IntuiText ITexta1 = {1,0,JAM2,10,2,&TOPAZ80,
 (UBYTE *)"ANIMATE",NULL};
struct Gadget Gadgeta55 = {&Gadgetb56,209,243,79,11,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Bordera1,NULL,&ITexta1,
 NULL,NULL,ANIMATE,NULL};
SHORT BorderVectorsa2[] = {0,0,80,0,80,12,0,12,0,0};
struct Border Bordera2 = {-1,-1,3,0,JAM1,5,
 BorderVectorsa2,NULL};
struct IntuiText ITexta2 = {1,0,JAM2,14,2,&TOPAZ80,
 (UBYTE *)"COLORS",NULL};
struct Gadget Gadgeta54 = {&Gadgeta55,125,243,79,11,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Bordera2,NULL,&ITexta2,
 NULL,NULL,COLORS,NULL};
USHORT Palette[] = {
 0x444,0x666,0x777,0x999,
 0xAAA,0xCCC,0xDDD,0xFFF,
 0x444,0x666,0x777,0x999,
 0xAAA,0xCCC,0xDDD,0xFFF};
USHORT ImageData1[] = {
0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,0x6004,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4001,0xF804,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4007,0xFE04,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4001,0xF804,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,0x6004,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0,0,
0,0,0,0,0,0,0,0,
0,0,0,2,0,2,0,2,
0,2,0,2,0,2,0,2,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFF
A,
0x3FFF,0x9FFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFF
A,
0x3FFF,0xFE7A,0x3000,0x1A,0x3FC0,0x7FFA,0x3FFE,0x7FA,
```

```
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFF3A,0x3FFF,0xE1FA,
0x33FF,0xFF9A,0x3F0E,0x1FFA,0x3FF8,0x1FA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0x3E00,0xFE7A,0x3FFE,0x1FFA,0x33FF,0xFF9A,
0x3E7F,0xCFFA,0x3FFE,0x7FA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,
0x3CFE,0x7CFA,0x3FE1,0xFFFA,0x33FF,0xFF9A,0x3E7F,0xCFFA,
0x3FFF,0x1FFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3BFF,0x3FA,
0x3E1F,0xFFFA,0x33FF,0xFF9A,0x3E7F,0xCFFA,0x3FFE,0x7FFA,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x37FF,0xFFFA,0x39FF,0xFFFA,
0x33FF,0xFF9A,0x3F0E,0x1FFA,0x3FFE,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3000,0x1A,
0x3FC0,0x7FFA,0x3F80,0x1FFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0,2,0,2,
0,2,0,2,0,2,0,2,
0,2,0,6,0,6,0,6,
0,6,0,0x6006,0,6,0,6,
0,6,0,0x186,0xFFF,0xFFE6,0x3F,0x8006,
1,0xF806,0,6,0,6,0,0xC6,
0,0x1E06,0xC00,0x66,0xF1,0xE006,7,0xFE06,
0xE4,0x3C06,0x93C,0xE186,0x1FF,0x186,1,0xE006,
0xC00,0x66,0x180,0x3006,1,0xF806,0x84,0x2406,
0x924,0x9246,0x301,0x8306,0x1E,6,0xC00,0x66,
0x180,0x3006,0,0xE006,0x84,0x3C06,0x924,0x9246,
0x400,0xFC06,0x1E0,6,0xC00,0x66,0x180,0x3006,
1,0x8006,0x84,0x2806,0x924,0x9246,0x800,6,
0x600,6,0xC00,0x66,0xF1,0xE006,1,6,
0xE7,0x2406,0xF24,0xE186,0,6,0,6,
0xFFF,0xFFE6,0x3F,0x8006,0x7F,0xE006,0,6,
0,6,0,6,0,6,0,6,
0,6,0,6,0,6,0,6,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image1 = {
 54,193,
 223,11,
 4,
 ImageData1,
 0xF,0,
 NULL};

struct Gadget Gadget54 = {&Gadgeta54,0,0,1.1,
 GADGHBOX¦GADGHIMAGE¦GADGIMAGE,
 NULL,BOOLGADGET,(APTR)&Image1,NULL,NULL,
 NULL,NULL,NULL,NULL};

struct Gadget Gadget53 = {&Gadget54,1,120,44,22,
 GADGHBOX¦GADGHIMAGE,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,NULL,NULL};

SHORT BorderVectors1[] = {0,0,80,0,80,12,0,12,0,0};
struct Border Border1 = {-1,-1,3,0,JAM1,5,BorderVectors1,
 NULL};

struct IntuiText IText1 = {1,0,JAM2,18,2,&TOPAZ80,
 (UBYTE *)"ABOUT",NULL};

struct Gadget Gadget52 = {&Gadget53,293,243,79,11,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border1,NULL,&IText1,
 NULL,NULL,ABOUT,NULL};

SHORT BorderVectors2[] = {0,0,80,0,80,12,0,12,0,0};
struct Border Border2 = {-1,-1,3,0,JAM1,5,BorderVectors2,
 NULL};

struct IntuiText IText2 = {1,0,JAM2,24,2,&TOPAZ80,
 (UBYTE *)"SAVE",NULL};

struct Gadget Gadget51 = {&Gadget52,209,228,79,11,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border2,NULL,&IText2,
 NULL,NULL,SAVE,NULL};

SHORT BorderVectors3[] = {0,0,80,0,80,12,0,12,0,0};
struct Border Border3 = {-1,-1,3,0,JAM1,5,BorderVectors3,
 NULL};

struct IntuiText IText3 = {1,0,JAM2,24,2,&TOPAZ80,
 (UBYTE *)"LOAD",NULL};

struct Gadget Gadget50 = {&Gadget51,125,228,79,11,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border3,NULL,&IText3,
 NULL,NULL,LOAD,NULL};

struct Gadget Gadget49 = {&Gadget50,245,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,UNDO,NULL};

struct Gadget Gadget48 = {&Gadget49,213,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,CLR,NULL};

struct Gadget Gadget47 = {&Gadget48,181,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,FILL,NULL};

struct Gadget Gadget46 = {&Gadget47,149,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,CIRC,NULL};

struct Gadget Gadget45 = {&Gadget46,117,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,RECT,NULL};

struct Gadget Gadget44 = {&Gadget45,53,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,DRAW,NULL};

struct Gadget Gadget43 = {&Gadget44,86,193,31,11,NULL,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,LINE,NULL};

SHORT BorderVectors4[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border4 = {-1,-1,1,0,JAM1,5,BorderVectors4,
 NULL};

struct Gadget Gadget42 = {&Gadget43,545,75,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border4,NULL,NULL,
 NULL,NULL,GADGETIMAGE,19};

SHORT BorderVectors5[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border5 = {-1,-1,1,0,JAM1,5,BorderVectors5,
 NULL};

struct Gadget Gadget41 = {&Gadget42,545,56,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border5,NULL,NULL,
 NULL,NULL,GADGETIMAGE,14};

SHORT BorderVectors6[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border6 = {-1,-1,1,0,JAM1,5,
 BorderVectors6,NULL};

struct Gadget Gadget40 = {&Gadget41,545,37,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border6,NULL,NULL,
 NULL,NULL,GADGETIMAGE,9};

SHORT BorderVectors7[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border7 = {-1,-1,1,0,JAM1,5,
 BorderVectors7,NULL};

struct Gadget Gadget39 = {&Gadget40,545,18,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border7,NULL,NULL,
 NULL,NULL,GADGETIMAGE,4};
```

```
SHORT BorderVectors8[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border8 = {-1,-1,1,0,JAM1,5,BorderVector
s8,
 NULL};

struct Gadget Gadget38 = {&Gadget39,508,75,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border8,NULL,NULL,
 NULL,NULL,GADGETIMAGE,18};

SHORT BorderVectors9[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border9 = {-1,-1,1,0,JAM1,5,BorderVector
s9,
 NULL};

struct Gadget Gadget37 = {&Gadget38,471,75,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border9,NULL,NULL,
 NULL,NULL,GADGETIMAGE,17};

SHORT BorderVectors10[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border10 = {-1,-1,1,0,JAM1,5,
 BorderVectors10,NULL};

struct Gadget Gadget36 = {&Gadget37,434,75,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border10,NULL,NULL,
 NULL,NULL,GADGETIMAGE,16};

SHORT BorderVectors11[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border11 = {-1,-1,1,0,JAM1,5,
 BorderVectors11,NULL};

struct Gadget Gadget35 = {&Gadget36,397,75,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border11,NULL,NULL,
 NULL,NULL,GADGETIMAGE,15};

SHORT BorderVectors12[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border12 = {-1,-1,1,0,JAM1,5,
 BorderVectors12,NULL};

struct Gadget Gadget34 = {&Gadget35,508,56,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border12,NULL,NULL,
 NULL,NULL,GADGETIMAGE,13};

SHORT BorderVectors13[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border13 = {-1,-1,1,0,JAM1,5,
 BorderVectors13,NULL};

struct Gadget Gadget33 = {&Gadget34,471,56,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border13,NULL,NULL,
 NULL,NULL,GADGETIMAGE,12};

SHORT BorderVectors14[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border14 = {-1,-1,1,0,JAM1,5,
 BorderVectors14,NULL};

struct Gadget Gadget32 = {&Gadget33,434,56,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border14,NULL,NULL,
 NULL,NULL,GADGETIMAGE,11};

SHORT BorderVectors15[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border15 = {-1,-1,1,0,JAM1,5,
 BorderVectors15,NULL};

struct Gadget Gadget31 = {&Gadget32,397,56,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border15,NULL,NULL,
 NULL,NULL,GADGETIMAGE,10};

SHORT BorderVectors16[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border16 = {-1,-1,1,0,JAM1,5,
 BorderVectors16,NULL};

struct Gadget Gadget30 = {&Gadget31,508,37,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border16,NULL,NULL,
 NULL,NULL,GADGETIMAGE,8};

SHORT BorderVectors17[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border17 = {-1,-1,1,0,JAM1,5,
 BorderVectors17,NULL};

struct Gadget Gadget29 = {&Gadget30,471,37,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border17,NULL,NULL,
 NULL,NULL,GADGETIMAGE,7};

SHORT BorderVectors18[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border18 = {-1,-1,1,0,JAM1,5,
 BorderVectors18,NULL};
```

In den reichsten Fischgründen Westafrikas:

Leere Netze

Vor der Küste Senegals hatten traditionsgemäß viele Kleinfischer ein sicheres Auskommen durch den Fang von Thunfisch, Sardinen und Krustentieren. Heute bevölkern sie die Slums von Dakar, der Hauptstadt des Senegal, oder arbeiten als schlechtbezahlte Tagelöhner auf den staatlich geförderten Hochseefangflotten, die ihre Fischgründe leerfischen. Eine kleine Gruppe junger Leute geht einen anderen Weg: Sie schlossen sich zu einer Fischereigenossenschaft zusammen. Durch Kredite aus BROT FÜR DIE WELT-Spenden können sie ihre Einbäume so ausstatten, daß sie trotz erdrückender Konkurrenz das Überleben ihrer Familien sichern.

BROT FÜR DIE WELT-Spenden helfen bei der Finanzierung.
Postf. 10 11 42 · 7000 Stuttgart 10

Brot für die Welt

Postgiro Köln 500500-500

QUALITÄT SETZT SICH DURCH

Computer Zubehör

DZ
Detlev Ziegler
Weidenstr. 29
4352 Herten 7
Tel. 02 09/61 13 93

Aus unserem SUPER Angebot

FRED FISH und KICKSTART jede Diskette	2,–
10 Disketten mit FONTS Benutzbar für alle Textprogramme	33,–
10 Disketten voller SPIELE z.B: MONOPOLY,RISIKO, u.v.a.	33,–
10 EROTIC Disketten mit prickelnden Bildern NUR GEGEN ALTERSNACHWEIS	33,–
TEXTVERARBEITUNG MS-TEXT *DEUTSCH*	4,–

12 DM Kosten unsere 4 DEUTSCHEN Katalogdisketten **EIN MUSS FÜR ALLE PD USER**

Alle Preise zuzüglich Versandkosten Vorkasse 6,–, Nachnahme 8,– Ausland nur gegen Vorkasse 10,–

3,5 ZOLL AB 2,–, 5,25 ZOLL AB 1,–

PD Preise

WIR FÜHREN ALLE GÄNGIGEN PD SERIEN z.B: FRED FISH, KICKSTART, ACS, KISS, POSEIDON, u.v.a. INSGESAMT CA. 3000 DISKETTEN AUS 42 SERIEN

BESTELLEN SIE SOFORT
02 09/61 13 93

NewTom GbR

HARDWARE, SOFTWARE UND ZUBEHÖR

Goethestraße 30 · 4100 Duisburg 18 (Walsum)
24-h-Bestellservice Telefon 02 03/49 57 97
Geschäftszeiten: Mo–Fr 16:00–18:30 Sa 10:00–14:00 Uhr

Leerdisketten
3,5" NoName 2DD 135 Tpi 50 Stück = 85,– 100 Stück = 169,–
3,5" EDIXA 2DD 135 Tpi 50 Stück = 105,– 100 Stück = 199,–

Laufwerke für AMIGA 500/1000/2000
3,5" WINNER Drive Slimline (nur 25,4 mm Höhe) 229,–
5,25" WINNER Drive 40/80 Track umschaltbar 269,–
alle Laufwerke mit durchgeführtem Bus, abschaltbar
3,5" Laufwerk für A2000 intern im Einbausatz 169,–

Speichererweiterungen
0,5 MByte WINNER Ramkarte für A500 m. Uhr, abschaltbar 209,–
8 MByte Ramkarte Commodore A2058 für AMIGA 2000 bestückt mit 2 MByte Ram 1098,–

Harddisks
20 MByte Harddisk COMMODORE A590 für Amiga 500 autobootend, intern auf 2 MByte RAM aufrüstbar
bestückt mit 0 MByte Ram 1035,–
bestückt mit 0,5 MByte Ram 1211,–
bestückt mit 1,0 MByte Ram 1385,–
bestückt mit 2,0 MByte Ram 1735,–

WINNER-Autoboot-Harddisk für AMIGA 500/1000
Gehäuse 330 mm x 330 mm x 60 mm, ideal als Monitoruntersatz
20 MB 988,– 31 MB 1088,– 40 MB 1268,– 63 MB 1598,–
WINNER Filecard für AMIGA 2000 mit Autobootkarte
20 MB 879,– 31 MB 968,– 40 MB 1198,– 47 MB 1298,–
WINNER-Autoboot-Einbauset für AMIGA 2000
30 MB 890,– 63 MB 1288,–
Alle Harddisks sind bereits unter FFS formatiert und mit WB 1.3d installiert, nach Einbau sofort betriebsbereit.
Weitere Harddisks, Drucker und Zubehör auf Anfrage

PUBLIC DOMAIN (Stand Dezember 1989)
Stückpreis 4,50 ab 10 = 3,50 ab 50 = 2,80 ab 100 = 2,50 DM
Fish –260, Kickstart –230, Taifun –120, Erotik –56, Auge –32
Lieferung per NN, Versandkosten 8 DM, Ausland Vorkasse

```
struct Gadget Gadget28 = {&Gadget29,434,37,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border18,NULL,NULL,
 NULL,NULL,GADGETIMAGE,6};

SHORT BorderVectors19[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border19 = {-1,-1,1,0,JAM1,5,
 BorderVectors19,NULL};

struct Gadget Gadget27 = {&Gadget28,397,37,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border19,NULL,NULL,
 NULL,NULL,GADGETIMAGE,5};

SHORT BorderVectors20[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border20 = {-1,-1,1,0,JAM1,5,
 BorderVectors20,NULL};

struct Gadget Gadget26 = {&Gadget27,508,18,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border20,NULL,NULL,
 NULL,NULL,GADGETIMAGE,03};

SHORT BorderVectors21[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border21 = {-1,-1,1,0,JAM1,5,
 BorderVectors21,NULL};

struct Gadget Gadget25 = {&Gadget26,471,18,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border21,NULL,NULL,
 NULL,NULL,GADGETIMAGE,02};

SHORT BorderVectors22[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border22 = {-1,-1,1,0,JAM1,5,
 BorderVectors22,NULL};

struct Gadget Gadget24 = {&Gadget25,434,18,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border22,NULL,NULL,
 NULL,NULL,GADGETIMAGE,01};

SHORT BorderVectors23[] = {0,0,33,0,33,17,0,17,0,0};
struct Border Border23 = {-1,-1,1,0,JAM1,5,
 BorderVectors23,NULL};

struct Gadget Gadget23 = {&Gadget24,397,18,32,16,
 GADGHBOX,
 RELVERIFY,BOOLGADGET,(APTR)&Border23,NULL,NULL,
 NULL,NULL,GADGETIMAGE,00};

struct Image Image2 = {0,0,20,10,0,NULL,0,0xF,NULL};

struct Gadget Gadget22 = {&Gadget23,25,103,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image2,NULL,NULL,
 NULL,NULL,GADGCOLOR,15};

struct Image Image3 = {0,0,20,10,0,NULL,0,0xE,NULL};

struct Gadget Gadget21 = {&Gadget22,25,91,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image3,NULL,NULL,
 NULL,NULL,GADGCOLOR,14};

struct Image Image4 = {0,0,20,10,0,NULL,0,0xD,NULL};

struct Gadget Gadget20 = {&Gadget21,25,79,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image4,NULL,NULL,
 NULL,NULL,GADGCOLOR,13};

struct Image Image5 = {0,0,20,10,0,NULL,0,0xC,NULL};

struct Gadget Gadget19 = {&Gadget20,25,67,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image5,NULL,NULL,
 NULL,NULL,GADGCOLOR,12};

struct Image Image6 = {0,0,20,10,0,NULL,0,0xB,NULL};

struct Gadget Gadget18 = {&Gadget19,25,55,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image6,NULL,NULL,
 NULL,NULL,GADGCOLOR,11};

struct Image Image7 = {0,0,20,10,0,NULL,0,0xA,NULL};

struct Gadget Gadget17 = {&Gadget18,25,43,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image7,NULL,NULL,
 NULL,NULL,GADGCOLOR,10};

struct Image Image8 = {0,0,20,10,0,NULL,0,9,NULL};

struct Gadget Gadget16 = {&Gadget17,25,31,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image8,NULL,NULL,
 NULL,NULL,GADGCOLOR,9};

struct Image Image9 = {0,0,20,10,0,NULL,0,8,NULL};

struct Gadget Gadget15 = {&Gadget16,25,19,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image9,NULL,NULL,
 NULL,NULL,GADGCOLOR,8};

struct Image Image10 = {0,0,20,10,0,NULL,0,7,NULL};

struct Gadget Gadget14 = {&Gadget15,1,103,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image10,NULL,NULL,
 NULL,NULL,GADGCOLOR,07};

struct Image Image11 = {0,0,20,10,0,NULL,0,6,NULL};

struct Gadget Gadget13 = {&Gadget14,1,91,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image11,NULL,NULL,
 NULL,NULL,GADGCOLOR,06};

struct Image Image12 = {0,0,20,10,0,NULL,0,5,NULL};

struct Gadget Gadget12 = {&Gadget13,1,79,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image12,NULL,NULL,
 NULL,NULL,GADGCOLOR,05};

struct Image Image13 = {0,0,20,10,0,NULL,0,4,NULL};

struct Gadget Gadget11 = {&Gadget12,1,67,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image13,NULL,NULL,
 NULL,NULL,GADGCOLOR,4};

struct Image Image14 = {0,0,20,10,0,NULL,0,3,NULL};

struct Gadget Gadget10 = {&Gadget11,1,55,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image14,NULL,NULL,
 NULL,NULL,GADGCOLOR,3};

struct Image Image15 = {0,0,20,10,0,NULL,0,2,NULL};

struct Gadget Gadget9 = {&Gadget10,1,43,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image15,NULL,NULL,
 NULL,NULL,GADGCOLOR,2};

struct Image Image16 = {0,0,20,10,0,NULL,0,1,NULL};

struct Gadget Gadget8 = {&Gadget9,1,31,20,10,
 GADGHBOX|GADGHIMAGE|GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image16,NULL,NULL,
 NULL,NULL,GADGCOLOR,1};

SHORT BlackBorder[] = {1,1,20,1,20,10,1,10,1,1};
struct Border Image17 = {-1,-1,1,0,JAM1,5,BlackBorder,
 NULL};

struct Gadget Gadget7 = {&Gadget8,1,19,20,10,
 GADGHBOX|GADGHIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image17,NULL,NULL,
 NULL,NULL,GADGCOLOR,0};

struct Gadget Gadget6 = {&Gadget7,34,0,507,12,
 GADGHBOX|GADGHIMAGE,
 RELVERIFY,BOOLGADGET,NULL,NULL,NULL,
 NULL,NULL,DRAGSCREEN,NULL};

SHORT BorderVectors24[] = {0,0,322,0,322,161,0,161,0,1};
struct Border Border24 = {-2,-1,3,0,COMPLEMENT,5,
 BorderVectors24,NULL};

struct Gadget Gadget5 = {&Gadget6,55,18,320,160,
 GADGHBOX|GADGHIMAGE,
 GADGIMMEDIATE|FOLLOWMOUSE,BOOLGADGET,
 (APTR)&Border24,NULL,NULL,
 NULL,NULL,DRAWGADG,NULL};

USHORT ImageData18[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0,2,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FF0,0x1A,
0x3FF0,0x1A,0x3FF0,0x1A,0x3FF0,0x1A,0x3FF0,0x1A,
0x3FFF,0xFFFA,0xFFFF,0xFFFE,0,2,0,6,
0xFFF,0xE006,0x800,0x2006,0x80F,0xFFE6,0x80F,0xFFE6,
0x80F,0xFFE6,0xFFF,0xFFE6,0xF,0xFFE6,0,6,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image18 = {0,0,31,11,4,ImageData18,0xF,0,
NULL};

USHORT ImageData19[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0xFFFF,0xFFFE,
0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xE018,
0xBFFF,0xE018,0xBFFF,0xE018,0xBFFF,0xE018,0xBFF0,0x18,
0xBFFF,0xFFF8,0x8000,0,0xFFFF,0xFFFE,0xC000,0,
0xCFFF,0xE000,0xC800,0x2000,0xC800,0x3FE0,0xC800,0x3FE0,
0xC800,0x3FE0,0xCFFF,0xFFE0,0xC00F,0xFFE0,0xC000,0,
0x8000,0,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};
```

```
struct Image Image19 = {0,0,31,11,4,ImageData19,0xF,0,
 NULL};

struct Gadget Gadget4 = {&Gadget5,576,1,31,11,
 GADGHIMAGE¦GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image18,(APTR)&Image19,
 NULL,
 NULL,NULL,SCREENBACK,NULL};

USHORT ImageData20[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0,2,
0x3FFF,0xFFFA,0x3000,0x1FFA,0x3000,0x1FFA,0x300F,0xFFFA,
0x300F,0xFFFA,0x300F,0xFFFA,0x300F,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0xFFFF,0xFFFE,0,2,0,6,
0xFFF,0xE006,0xFFF,0xE006,0xFFF,0xFFE6.0xFF8,0x26,
0xFF8,0x26,0xFF8,0x26,0xF,0xFFE6,0,6,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image20 = {0,0,31,11,4,ImageData20,0xF,0,
 NULL};

USHORT ImageData21[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0xFFFF,0xFFFE,
0xBFFF,0xFFF8,0xB000,0x1FF8,0xB000,0x1FF8,0xB000,0x1FF8,
0xB000,0x1FF8,0xB000,0x1FF8,0xB000,0x1FF8,0xBFFF,0xFFF8,
0xBFFF,0xFFF8,0x8000,0,0xFFFF,0xFFFE,0xC000,0,
0xCFFF,0xE000,0xCFFF,0xE000,0xCFFF,0xFFE0,0xCFFF,0xE020,
0xCFFF,0xE020,0xCFFF,0xE020,0xC00F,0xFFE0,0xC000,0,
0x8000,0,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image21 = {0,0,31,11,4,ImageData21,0xF,0,
 NULL};

struct Gadget Gadget3 = {&Gadget4,610,1,31,11,
 GADGHIMAGE¦GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image20,(APTR)&Image21,
 NULL,
 NULL,NULL,SCREENFRONT,NULL};

USHORT ImageData22[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0,2,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0xFFFF,0xFFFE,0,2,0,6,
0xFE0,6,0x820,6,0xFFF,0xFFE6,0x20,0x26,
0x20,0x26,0x20,0x26,0x3F,0xFFE6,0,6,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image22 = {0,0,31,11,4,ImageData22,0xF,0,
 NULL};

USHORT ImageData23[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0xFFFF,0xFFFE,
0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,
0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,
0xBFFF,0xFFF8,0x8000,0,0xFFFF,0xFFFE,0xC000,0,
0xCFE0,0,0xC820,0,0xCFFF,0xFFE0,0xC020,0x20,
0xC020,0x20,0xC020,0x20,0xC03F,0xFFE0,0xC000,0,
0x8000,0,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image23 = {0,0,31,11,4,ImageData23,0xF,0,
 NULL};

struct Gadget Gadget2 = {&Gadget3,542,1,31,11,
 GADGHIMAGE¦GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image22,
 (APTR)&Image23,NULL,
 NULL,NULL,ICONIZE,NULL};

USHORT ImageData24[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0,2,
0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,0x3FF0,0x1FFA,
0x3FF3,0x9FFA,0x3FF0,0x1FFA,0x3FFF,0xFFFA,0x3FFF,0xFFFA,
0x3FFF,0xFFFA,0xFFFF,0xFFFE,0,2,0,6,
0xFFF,0xFFE6,0x800,0x26,0x80F,0xE026,0x80C,0x6026,
0x80F,0xE026,0x800,0x26,0xFFF,0xFFE6,0,6,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image24 = {0,0,31,11,4,ImageData24,0xF,0,
 NULL};

USHORT ImageData25[] = {
0,0,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0x4000,4,0x4000,4,
0x4000,4,0x4000,4,0,0,0xFFFF,0xFFFE,
0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,0xBFF0,0x1FF8,
0xBFF3,0x9FF8,0xBFF0,0x1FF8,0xBFFF,0xFFF8,0xBFFF,0xFFF8,
0xBFFF,0xFFF8,0x8000,0,0xFFFF,0xFFFE,0xC000,0,
0xCFFF,0xFFE0,0xC800,0x20,0xC80F,0xE020,0xC80C,0x6020,
0xC80F,0xE020,0xC800,0x20,0xCFFF,0xFFE0,0xC000,0,
0x8000,0,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,
0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE,0xFFFF,0xFFFE};

struct Image Image25 = {0,0,31,11,4,ImageData25,0xF,0,
 NULL};

struct Gadget Gadget1 = {&Gadget2,0,1,31,11,
 GADGHIMAGE¦GADGIMAGE,
 RELVERIFY,BOOLGADGET,(APTR)&Image24,(APTR)&Image25,
 NULL,
 NULL,NULL,CLOSESPRED,NULL};

#define GadgetList1 Gadget1

struct NewWindow NewWindowStructure1 = {
 0,0,640,256,0,1,
 GADGETDOWN¦GADGETUP¦MOUSEMOVE¦MOUSEBUTTONS¦VANILLAKEY,
 BORDERLESS¦ACTIVATE¦RMBTRAP¦NOCAREREFRESH¦REPORTMOUSE,
 &Gadget1,NULL,NULL,NULL,NULL,5,5,-1,-1,
 CUSTOMSCREEN};
```

Listing: SprEd.h

```
************************************************************
*                    AMIGA DOS INFO                         *
*  Programmname       : Defines.h                           *
*  Funktion           : Labels für SprEd.c                  *
*  Sprache            : Aztec-C                             *
*  Besonderheiten     : keine                               *
************************************************************

#define DRAGSCREEN 1
#define ICONIZE  2
#define SCREENFRONT 3
#define SCREENBACK 4
#define CLOSESPRED 5
#define ABOUT  6
#define SAVE  7
#define LOAD  8
#define UNDO  9
#define CLR   10
#define FILL  11
#define CIRC  12
#define RECT  13
#define LINE  14
#define DRAW  15
#define DRAWGADG 16
#define COLORS  17
#define ANIMATE  18
#define MAKESRC  19
#define GADGCOLOR 20
#define GADGETIMAGE 21
```

Listing: Defines.h

```
************************************************************
*                    AMIGA DOS INFO                         *
*  Programmname       : SprEd.c                             *
*  Funktion           : Hauptprogramm                       *
*  Sprache            : Aztec-C                             *
*  Besonderheiten     : benötigt die Arp-Library            *
************************************************************

/*******************************************************************\
 *                        Sprite Edit                              *
 *******************************************************************
 *                Written 1989 by G. Glendown                      *
```

```
\**********************************************************
*****************/

#include "defines.h"
#include "spred.h"
#include "arpbase.h"
#include "arpfunctions.h"

#define ScrDef NewScreenStructure
#define WinDef NewWindowStructure1

struct IntuitionBase *IntuitionBase;
struct GfxBase *GfxBase;
struct ArpBase *ArpBase;
VOID *OpenLibrary();

UBYTE image[20][16][16];
UBYTE undo[16][16];

struct Screen *scr,*OpenScreen();
struct Window *win,*OpenWindow();

struct IntuiMessage *msg,*GetMsg();

struct UCopList *myCop,*origCop;
struct View *view,*ViewAddress();
struct ViewPort *vp;
struct RastPort *rp;

BOOL SCREENMOVED;

long class;
int code;

VOID *AllocMem();

USHORT SecPal[] = {
 0x0222,0x0FFF,0x0F00,0x000F,
 0x00F0,0x0484,0x04F8,0x0CC0,
 0x0080,0x0048,0x000F,0x004C,
 0x0808,0x0844,0x0FF0,0x00FF};

char *DoFile();
char fname[256];

BOOL DRAWIT,ERASE;

int CurCol,CurImage;
long CIx,CIy;

struct FileHandle *fh,*Open();
long Read(),Write();

main()
{
int t;
struct Gadget *gad;
char *c;
long l;

 SCREENMOVED=FALSE;
 CurCol=1;
 CurImage=0;
 CIx=397L;
 CIy=18L;
 DRAWIT=FALSE;
 ERASE=FALSE;
 OpenStuff();

 for (t=0;t==0;) {
  WaitPort(win->UserPort);
  msg=GetMsg(win->UserPort);
  class=msg->Class;
  code=msg->Code;

  switch (class) {
   case MOUSEBUTTONS:

    switch (code) {

     case SELECTUP:
      DRAWIT=FALSE;
      break;

     case MENUDOWN:
      DRAWIT=TRUE;
      ERASE=TRUE;
      CopyToUndo(CurImage);
      Pressed(msg->MouseX,msg->MouseY);
      break;

     case MENUUP:
      DRAWIT=FALSE;
      ERASE=FALSE;
      break;
    }
    break;

   case GADGETUP:
    gad=(struct Gadget *)msg->IAddress;

    switch (gad->GadgetID) {

     case CLOSESPRED:
      t=1;
      break;

     case SCREENBACK:
      ScreenToBack(scr);
      break;

     case SCREENFRONT:
      ScreenToFront(scr);
      break;

     case DRAGSCREEN:
      if (SCREENMOVED)
       MoveScreen(scr,0L,-128L),
       SCREENMOVED=FALSE;
      else
       MoveScreen(scr,0L,128L),
       SCREENMOVED=TRUE;
      break;

     case ABOUT:
#ifdef PART2
      DisplayAbout();
#endif
      break;

     case SAVE:
      c=DoFile((UBYTE *)"File sichern:",fname);
      if (c) {
       fh=Open(fname,MODE_NEWFILE);
       if (fh)
        l=Write(fh,"SE10",4L),
        l+=Write(fh,image,5120L),
        Close(fh);
       if (l!=5124L)
#ifndef PART2
        DisplayBeep(scr),
        Delay(10L),
        DisplayBeep(scr);
#else
        ShowError("Fehler beim Speichern!");
#endif
      }
      break;

     case LOAD:
      c=DoFile((UBYTE *)"File laden:",fname);

      if (c) {
       fh=Open(fname,MODE_OLDFILE);

       if (fh) {
        Read(fh,image,4L);
        if (strncmp("SE10",(UBYTE *)image,4)==0) {

         if (Read(fh,image,5120L)==5120L) {
          for (t=0;t<20;t++)
           DisplaySImage(t);
          DisplayImage(CurImage);
         }
#ifdef PART2
         else ShowError("Lese-Fehler!");
#endif
        }
#ifdef PART2
        else ShowError("Kein SprEd-Datenfile!");
#endif
        Close(fh);
       }
       CopyToUndo(CurImage);
       t=0;
      }
      break;

     case UNDO:
      CopyFromUndo(CurImage);
      Display(CurImage);
      break;

     case CLR:
      CopyToUndo(CurImage);
      {
      register int a,b;

       for (b=0;b<16;b++)
        for (a=0;a<16;a++)
         image[CurImage][a][b]=0;
      }
      Display(CurImage);
      break;

     case FILL:
#ifdef PART2
      DoFill();
#endif
      break;

     case CIRC:
#ifdef PART2
      DoCirc();
#endif
      break;

     case RECT:
#ifdef PART2
      DoRect();
#endif
      break;

     case LINE:
#ifdef PART2
      DoLine();
```

```
#endif
        break;

       case DRAW:
#ifdef PART2
        DoDraw();
#endif
        break;

       case DRAWGADG:
        DRAWIT=FALSE;
        break;

       case COLORS:
#ifdef PART2
        Palette();
#endif
        break;

       case ANIMATE:
#ifdef PART2
        DoAnimate();
#endif
        break;
       case MAKESRC:
#ifdef PART2
        CreateSrc();
#endif
        break;

       case GADGCOLOR:
        CurCol=(int)((long)(gad->UserData));

        if (CurCol!=0)
         SetAPen(rp,(long)CurCol),
         RectFill(rp,1L,120L,45L,142L);
        else {
         SetAPen(rp,1L),
         RectFill(rp,1L,120L,45L,142L);
         SetAPen(rp,0L),
         RectFill(rp,2L,121L,44L,141L);
        }
        break;

       case GADGETIMAGE:
        t=CurImage;
        CurImage=(int)((long)(gad->UserData));
        CIx=(long)(gad->LeftEdge);
        CIy=(long)(gad->TopEdge);
        if (t!=CurImage) DisplayImage(CurImage);
        t=0;
        CopyToUndo(CurImage);
        break;
      }
      break;

    case GADGETDOWN:
     gad=(struct Gadget *)msg->IAddress;

     switch (gad->GadgetID) {
      case DRAWGADG:
       CopyToUndo(CurImage);
       DRAWIT=TRUE;
       Pressed(msg->MouseX,msg->MouseY);
       break;
     }
     break;

    case MOUSEMOVE:
     if (DRAWIT) {
      Pressed(msg->MouseX,msg->MouseY);
     }
#ifdef PART2
     else HandleMovement(msg->MouseX,msg->MouseY);
#endif
     break;

    case VANILLAKEY:

     switch(toupper(code)) {
      case 'U':
       CopyFromUndo(CurImage);
       Display(CurImage);
       break;
     }
     break;

    default:
     printf("Class: %01x\n",class);
     break;
   }
   ReplyMsg(msg);
  }
  ex();
}

/**********************************************************************\
 *                         Image anzeigen                             *
\**********************************************************************/

Display(i)
int i;
{
 DisplayImage(i);
 DisplaySImage(i);
}

/**********************************************************************\
 *                   Image im Undo-Puffer abspeichern                 *
\**********************************************************************/

CopyToUndo(i)
int i;
{
register int a,b;

 for (b=0;b<16;b++)
  for (a=0;a<16;a++)
   undo[a][b]=image[i][a][b];
}

/**********************************************************************\
 *      Image aus Undo-Puffer holen, aktuelles Bild ab speichern      *
\**********************************************************************/

CopyFromUndo(i)
int i;
{
register int a,b;
register UBYTE r;

 for (b=0;b<16;b++)
  for (a=0;a<16;a++)
   r=undo[a][b],
   undo[a][b]=image[i][a][b],
   image[i][a][b]=r;
}

/**********************************************************************\
 *                   Image im Kleinformat anzeigen                    *
\**********************************************************************/

DisplaySImage(i)
int i;
{
register long x,y,a,b;
UBYTE c;

 x=(long)(397+(i%5)*37);
 y=(long)(18+(i/5)*19);
 SetAPen(rp,0L);
 RectFill(rp,x,y,x+31L,y+15L);

 for (b=0;b<16;b++)
  for (a=0;a<16;a++)

   if (c=image[i][a][b])
    SetAPen(rp,(long)c),
    WritePixel(rp,x+(a<<1),y+b),
    WritePixel(rp,x+(a<<1)+1L,y+b);
}

/**********************************************************************\
 *                    Image im Großformat anzeigen                    *
\**********************************************************************/

DisplayImage(i)
int i;
{
register int x,y;
UBYTE c;
 SetAPen(rp,0L);
 RectFill(rp,55L,18L,374L,177L);

 for (y=0;y<16;y++)

  for (x=0;x<16;x++)
   if (c=image[i][x][y])
    PPoint(x,y,c);
}

/**********************************************************************\
 *                   Abfrage der Maus-Koordinaten                     *
\**********************************************************************/

Pressed(x,y)
SHORT x,y;
{
 if ((x>=55)&&(x<375)) {
  if ((y>=18)&&(y<178)) {

   if (ERASE)
    PlotIt(x,y,0);
   else
    PlotIt(x,y,CurCol);
  }
 }
}
```

```
/*****************************************************
*****************\
 *                          Umrechnen Maus -> Image
                  *
\*****************************************************
*****************/

PlotIt(x,y,c)
int x,y,c;
{
 x-=55;
 y-=18;
 x/=20;
 y/=10;
 PlotPoint(x,y,c);
 image[CurImage][x][y]=(UBYTE)c;
}

/*****************************************************
*****************\
 *                      Punkt plotten (groß und klein)
                  *
\*****************************************************
*****************/

PlotPoint(x,y,c)
int x,y,c;
{
long ox,oy,ux,uy;
 ox=(long)(55+20*x);
 oy=(long)(18+10*y);
 ux=(long)(74+20*x);
 uy=(long)(27+10*y);

 SetAPen(rp,(long)c);
 WritePixel(rp,(long)(CIx+(x<<1)),(long)(CIy+y));
 WritePixel(rp,(long)(CIx+(x<<1)+1),(long)(CIy+y));

 RectFill(rp,ox,oy,ux,uy);
}

/*****************************************************
*****************\
 *                          Punkt plotten (nur groß)
                  *
\*****************************************************
*****************/

PPoint(x,y,c)
int x,y,c;
{
long ox,oy,ux,uy;
 ox=(long)(55+20*x);
 oy=(long)(18+10*y);
 ux=(long)(74+20*x);
 uy=(long)(27+10*y);

 SetAPen(rp,(long)c);
 RectFill(rp,ox,oy,ux,uy);
}

/*****************************************************
*****************\
 *   Resourcen holen (Libs, Screen, Window) , Copper i
nitialisieren    *
\*****************************************************
*****************/

OpenStuff()
{
int t;
 if (!(GfxBase=OpenLibrary("graphics.library",0L)))
  puts("GFX!"),ex();

 if (!(IntuitionBase=OpenLibrary("intuition.library",0
L)))
  puts("Intui"),ex();

 if (!(ArpBase=OpenLibrary("arp.library",34L)))
  puts("Ich brauche die Arp-Library!"),ex();

 if (!(scr=OpenScreen(&ScrDef)))
  puts("Scr"),ex();

 WinDef.Screen=scr;
 if (!(win=OpenWindow(&WinDef)))
  puts("Win"),ex();

 if (!(myCop=AllocMem((long)sizeof(struct UCopList),
  MEMF_PUBLIC|MEMF_CLEAR)))
   puts("Mem"),ex();

 view=ViewAddress();
 vp=view->ViewPort;
 rp=&(scr->RastPort);
 SetAPen(rp,1L),
 RectFill(rp,1L,120L,45L,142L);
 origCop=vp->UCopIns;
 vp->UCopIns=myCop;

 CWAIT(myCop,0L,0L);
 CMOVE(myCop,custom.color[0],(long)0xfff);
 CWAIT(myCop,1L,0L);
 CMOVE(myCop,custom.color[0],(long)0x000);

 for (t=1;t<16;t++)
  CMOVE(myCop,custom.color[t],(long)(Palette[t]));

 CWAIT(myCop,13L,0L);
 CMOVE(myCop,custom.color[0],(long)0xFFF);
 CWAIT(myCop,14L,0L);

 for (t=0;t<16;t++)
  CMOVE(myCop,custom.color[t],(long)SecPal[t]);

 CWAIT(myCop,185L,0L);
 CMOVE(myCop,custom.color[0],(long)0xfff);
 CWAIT(myCop,186L,0L);
 CMOVE(myCop,custom.color[0],0L);

 for (t=1;t<16;t++)
  CMOVE(myCop,custom.color[t],(long)(Palette[t]));

 CEND(myCop);
 RethinkDisplay();
}

/*****************************************************
*****************\
 *                          Beenden von SprEd
                  *
\*****************************************************
*****************/

ex()
{
 if (myCop)
  vp->UCopIns=origCop,
  RethinkDisplay(),
  FreeMem(myCop,(long)sizeof(struct UCopList));

 if (win) CloseWindow(win);
 if (scr) CloseScreen(scr);
 if (ArpBase) CloseLibrary(ArpBase);
 if (GfxBase) CloseLibrary(GfxBase);
 if (IntuitionBase) CloseLibrary(IntuitionBase);
 exit(0L);
}

/*****************************************************
*****************\
 *                       File-Requester-Aufruf
                  *
\*****************************************************
*****************/

struct FileRequester MyFileReq =
{
 NULL,
 NULL,
 NULL,
 NULL,
 0,
 0,
 NULL,
 NULL
} ;


char *DoFile(UserText,Path)
char *UserText,*Path;
{
char Dir[256],File[33],*Result;
int t,g;

 MyFileReq.fr_Window=win;

 for (t=strlen(Path);t>=0;t--)
  if ((Path[t]=='/')||(Path[t]==':')) break;

 if (Path[t]=='/')
  t--; /* Wenn '/', dann ein Zeichen weniger... */

 for (g=t;g>=0;Dir[g]=Path[g],g--); /* mal sehen ... *
/
 Dir[t+1]=0;         /* Ende */
 if (Path[t+1]=='/') t++; /* Und wieder eins mehr... *
/
 strncpy(File,&Path[t+1],32);

 if (UserText)
  MyFileReq.fr_Hail=UserText ;
 MyFileReq.fr_Dir=Dir ;
 MyFileReq.fr_File=File ;

 Result = (char *)FileRequest(&MyFileReq) ;

 if (Result)
 {
  if (strlen(File)>0) { /* got a file ? */
   strcpy(Path,Dir) ;

   TackOn(Path,File) ;

   return(Path) ;
  }
 }
 else return(NULL) ;
}

```

Listing: SprEd.c



AUGEN AUF BEIM DRUCKERKAUF

Welcher Drucker zu welchem Zweck? Eine kleine Marktübersicht

Nachdem wir uns im letzten Heft unter den Zweitlaufwerken umgeschaut haben, wollen wir diesmal ein weiteres wichtiges Peripheriegerät unter die Lupe nehmen: Den Drucker.

Einen Drucker zu kaufen, stellt so manchen frischgebackenen Amiga-Besitzer vor immense Schwierigkeiten. Drucker en masse – so könnte man ohne Umschweife die Marktsituation beschreiben. Doch welcher ist der richtige, welcher der am besten einsetzbare Drucker? Hierzu ein paar Hilfestellungen.

Vor dem Kauf lieber vorführen lassen – das erspart Ärger

Am besten macht man sich vor dem Kauf Gedanken darüber, was der zukünftige Schönschreiber können und leisten soll, und welche Anforderungen im Bezug auf die Hardware zu stellen sind. Zum zweiten Punkt sind eigentlich keine größeren Überlegungen notwendig; der Amiga verfügt von Hause aus über eine Parallelschnittstelle, die dem normalen Standard, wie er auch zum Beispiel bei PCs vorhanden ist, entspricht. Vorhanden sind dort acht Leitungen, welche die Zeichen, die vom Rechner zum Drucker geschickt werden, auf die acht Leitungen verteilen, also parallel senden, sowie diverse Steuerleitungen, über die unser Amiga die interne Kommunikation mit dem Drucker führt. Bei den meisten Druckern ist ebenfalls eine parallele Schnittstelle vorhanden, die sogenannte Centronics-Schnittstelle.

Einige Drucker verfügen direkt oder optional über serielle Schnittstellen; sie können zwar ebenfalls betrieben werden, jedoch sind sie auf dem Markt eindeutig in der Minderzahl.

Bleibt also nur noch, die Anforderungen an unseren Wunschdrucker zu notieren:

Verschiedene Druckertypen sind auf dem Markt zu finden: Matrixdrucker, Typenraddrucker, Tintenstrahldrucker und Laserdrucker.

Nadeln, Typenrad, Tintendüsen, Laserstrahl – alles vorhanden!

Der Matrixdrucker setzt Zeichen aus Punkten zusammen, die mittels Nadeln auf das Papier gebracht werden. Je mehr Nadeln ein Drucker hat, desto besser ist die Druckqualität, desto höher aber auch der Preis. Die heutigen, gängigsten Matrixrucker trennen sich in '9-Nadel-' beziehungsweise '24-Nadel-Drucker'. Teilweise gibt es schon 48-Nadel-Drucker, diese sind jedoch für den Hausgebrauch schlichtweg zu teuer (was allerdings nicht ewig gelten muß, siehe 24-Nadler).

Typenraddrucker besitzen an Stelle der Nadeln ein Typenrad, ähnlich wie eine Schreibmaschine, dessen Zeichen an die richtige Position gedreht wird. Die Schrift entspricht damit fast der Schreibmaschinenqualität. Leider besitzen Sie im Gegensatz zu Matrixdruckern einige Nachteile, darunter ist Grafikausdruck fast nicht möglich und die Geschwindigkeit ist nicht sehr hoch. Auch Schreibmaschinen mit Centronics-Schnittstellen sind hier zu nennen, allerdings kann es hier zu Problemen mit der Schnittstelle kommen.

Tintenstrahldrucker besitzen winzig kleine Düsen, durch die Tinte in hoher Geschwindigkeit auf das Papier 'geschossen' wird. Sie sind recht teuer und für den Heimgebrauch am Amiga zu unrentabel, da hier auch Spezialpapier benutzt werden muß.

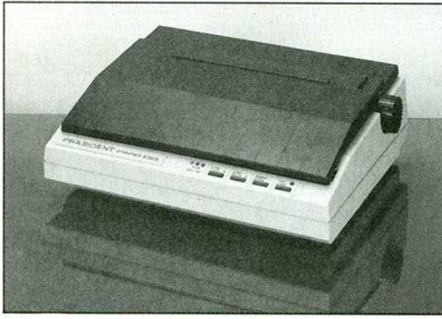
Laserdrucker haben von allen Druckern das beste Schriftbild, sind aber auch nicht gerade billig (wobei billig und preiswert relative Begriffe sind). Außerdem sind sie sehr wartungsintensiv und müssen ziemlich 'verhätschelt' werden. Sie sind aber eigentlich der Traum eines jeden Computerbesitzers, denn Texte, mit Grafiken versehen, auf Laserdrucker auszugeben, hat etwas mystisches an sich, denn viele Leute denken automatisch an Desktop Publishing und die immensen Möglichkeiten.

Bleiben wir jedoch lieber auf dem Boden der Tatsachen und schauen wir uns unter den Matrixdruckern um, die wohl eher unserem Geldbeutel zuträglich sind.

Anfoderungen an einen imaginären Drucker – Was muß er leisten und was können?

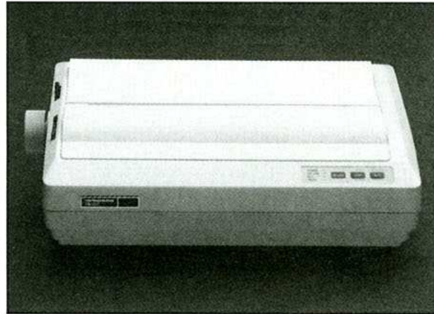
Bei Matrixdruckern kommt es vor allem darauf an, was man mit dem Drucker machen will. "Der Amiga ist ein Grafikcomputer!" – damit steht fest, daß der Ausdruck dieser Grafiken praktisch vorprogrammiert (wie sinnig!) ist. "Der Amiga ist ein Farbwunder!" – Eine Feststellung, die für die meisten Drucker nicht gilt, wenn...

Ja, wenn sie nicht an sich Farbdrucker sind. Deshalb sollte man sich überlegen, ob der Einsatz eines Farbdruckers lohnt, denn viele Programme haben die Steuer-



Name: *Präsident Printer 6325*
Preis: *DM 349, –*
Typ: *9-Nadel-Drucker*
Zeichensätze: *ASCII/IBM/Zusätzliche*
Puffer: *keine Angaben*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Innenliegend*
Besonderheiten: *Verschiedene Schnittstellenmodule erhältlich*

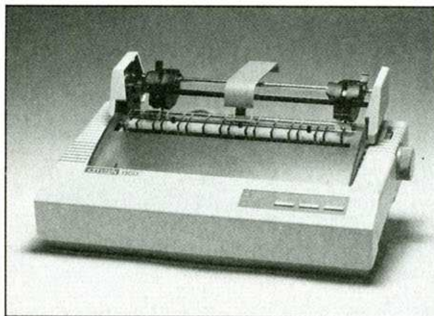
Name: *Mannesmann Tally MT-81*
Preis: *DM 399, –*
Typ: *9-Nadel-Drucker*
Zeichensätze: *ASCII/IBM/EPSON*
Puffer: *8 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch, Englisch*
DIP-Schalter: *Innenliegend*
Besonderheiten: *Serielle Schnittstelle als Zubehör Puffer auf 32 kByte erweiterbar*

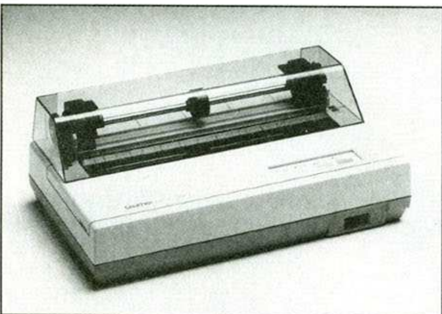




Name: *Star LC 10*
Preis: *DM 578, –*
Typ: *9-Nadel-Drucker*
Zeichensätze: *ASCII/IBM 1 und 2*
Puffer: *4 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *innenliegend*
Besonderheiten: *Einzelblatteinzug mit Papiermagazin als Zubehör.*
Star LC 10 Colour (Farbdrucker) 748, –

Name: *CITIZEN LSP 120D*
Preis: *DM 598, –*
Typ: *9-Nadel-Drucker*
Zeichensätze: *ASCII/IBM 1 und 2/ Internationale*
Puffer: *8 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Außenliegend*
Besonderheiten: *Serielle Schnittstelle optional*





Name: *Brother M 1109*
Preis: *DM 599, –*
Typ: *9-Nadel-Drucker*
Zeichensätze: *ASCII/IBM-Grafik/Internationale*
Puffer: *2 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Innenliegend*
Besonderheiten: *Rollenpapierhalter für DM 29, –*

Name: *Seikosha SL-80 IP*
Preis: *DM 799, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM 1 und 2/10 Zusätzliche*
Puffer: *16 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Außenliegend*
Besonderheiten: *Carbonfarbband, automatischer Einzelblatteinzug optional*



codes für Farbdruck nicht in ihrem Repertoire. Deshalb kann es schnell zum großen Frust kommen, wenn der Farbdrucker das 'Farb' einfach wegläßt und nur zum 'Drucker' wird. Zwar gibt es in der Public Domain jede Menge Druckertreiber, doch hat man nie den, den man gerade braucht.

Andererseits hat ein Farbausdruck, so es funktioniert, etwas Erhabenes an sich, und Bilder auf Farbdruckern ausgegeben, machen schon etwas her. Entscheiden Sie sich...

Sind Sie ein Listing-Großerzeuger? Und brauchen Sie Ihren Drucker nur zum Ausdruck Ihrer Listingseiten? Texte und Grafiken kommen bei Ihnen nur ab und zu vor und Sie haben keinen großen Anspruch an die Qualität? Dann sind Sie mit den 9-Nadel-Druckern gut bedient. Vor allem deshalb, weil die heutigen 9-Nadler auch qualitativ einiges zu bieten haben. NLQ heißt hier das Zauberwort – es steht für 'Near Letter Quality', einer Schrift, die der Qualität von Schreibmaschinen nahe kommt. Dazu besitzen 9-Nadler einige Schriftarten, mit denen sich schon eine anspruchsvollere Textgestaltung herstellen läßt.

Die höherwertigeren, aber auch preislich höherliegenden 24-Nadel-Drucker sind vor allem für Druckerfans interessant, die Wert auf schöne Schrift und guten Grafikausdruck legen. War es bei den 9-Nadlern 'NLQ', so heißt die Losung jetzt 'LQ' (Letter Quality). Je besser die Schrift, desto höher die Druckqualität. Dazu gehören Schriftarten wie 'Roman' oder 'Sans Serif' genauso wie Sonderschriftarten wie Schattenschrift oder 'Outline'. Sie finden am Amiga die weiteste Verbreitung, denn hier paßt der Deckel auf den Topf! Textverarbeitungen, wie das in diesem Heft vorgestellte UBM-Text oder unsere Textverarbeitung 'ConText-Amiga' aus Heft 1/90 nutzen die verschiedenen Schriftarten voll aus (Sehen Sie dazu auch den Artikel "Context lernt drucken" auf Seite 29 in diesem Heft). Die Preise für 24-Nadel-Drucker bewegen sich dabei von erschwinglich bis teuer. Auch hier heißt es wählen.

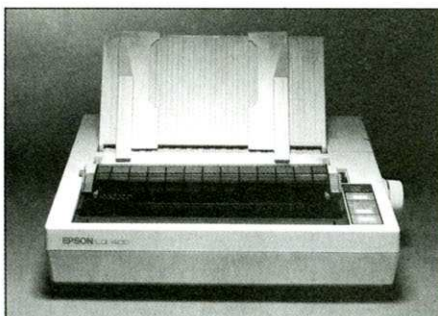
Eine Checkliste vor dem Druckerkauf ist notwendig, denn Sie wollen ja garantiert

nicht die berühmte Katze im Sack bekommen. Deshalb sollte unsere Checkliste folgende Punkte aufweisen:

Bevor man kauft sollte man ausprobieren – Hinweise zum Erwerb

○ Ist ein entsprechender Druckertreiber vorhanden (in 'Preferences' oder in den Programmen)? Wenn nein, läßt sich der Drucker mit anderen Druckertreibern installieren?
○ Verfügt der Drucker über verschiedene Zeichensätze (ASCII/IBM)?
○ Schreiben Sie viele Texte? Wenn ja, sollte der Drucker über mehrere Schriftarten verfügen, oder über diese zumindest optional, z.B. durch Steckkarten, verfügen.
○ Wieviel Pufferspeicher hat Ihr ausgewählter Drucker? Der Amiga ist zwar multitasking-fähig, jedoch behindert ein zu kleiner Speicher im Drucker problemloses Arbeiten. Drucker mit 8 kByte RAM sind wesentlich effektiver als Drucker mit 2 kByte.
○ Ist die Bedienung des Druckers einfach durchzuführen? Ist das Bedienfeld leicht zu erreichen und zu betätigen?
○ Und jetzt einer der wichtigsten Faktoren: das Handbuch. Ist ein deutsches Handbuch vorhanden? Billig-Angebote sind meistens Grauimporte und verfügen nur über englische Versionen des Handbuches.
○ Ist das Zubehör oder die Ersatzteile leicht zu beschaffen? Wie sieht es mit dem Service aus, wenn das Gerät mal defekt ist?
Wenn Sie diese Fragen für sich zufriedenstellend beantworten können, dann lassen Sie sich bei Fachhändlern noch einmal eingehend beraten – und lassen Sie sich das Gerät vorführen. Nur dann haben Sie Gewißheit, keine 'Sackkatze' nach Hause zu schleppen.
Um Ihnen beim Druckerkauf ein paar Anhaltspunkte mitzugeben, stellen wir Ihnen hier ein paar Drucker der niedrigen bis mittleren Preisklasse vor. Natürlich können wir nicht alle vorhandenen Drucker zeigen, die nebenstehenden jedoch bieten einen Einblick in den vorhandenen Markt.

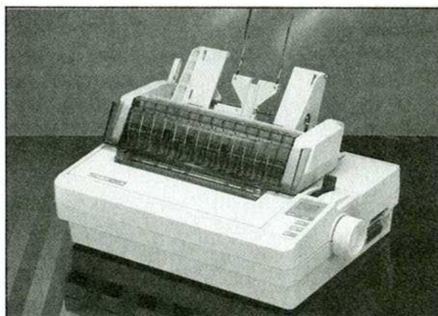
(jb)



Name: *EPSON LQ-400*
Preis: *DM 799, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM-Grafik/Internationale*
Puffer: *8 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Innenliegend*
Besonderheiten: *Schacht für externe ZeichensatzModule, Einzelblatteinzug optional*

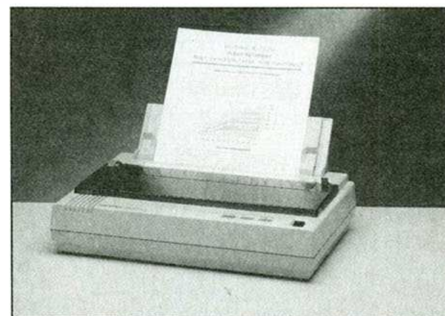
Name: *Star LC 24-10*
Preis: *DM 898, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM 1 und 2*
Puffer: *7 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Innenliegend*
Besonderheiten: *Lieferbares Zubehör: Einzelblatteinzug mit Papiermagazin, Schriftsteckkarten, batteriegepufferte 32 kByte RAM-Steckkarte*

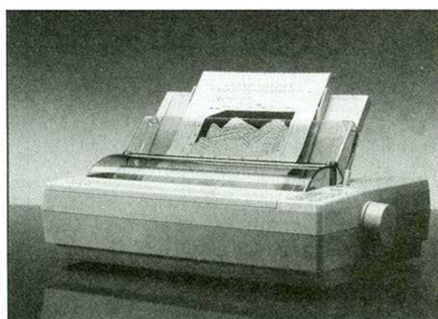




Name: *Citizen Swift 24*
Preis: *DM 1098, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM/Internationale*
Puffer: *8 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *ersetzt durch Tastenfeld*
Besonderheiten: *Lieferbares Zubehör: Farb- und Speichererweiterung, Font-Card Papiereinzug autom./manuell*

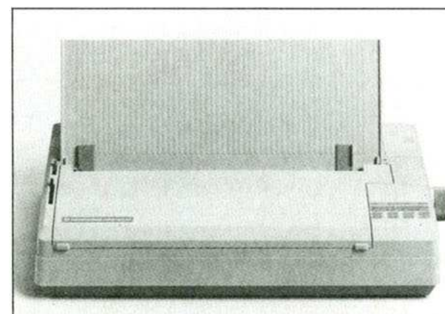
Name: *Brother M-1224 L*
Preis: *DM 1139, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM-Grafik/Internationale*
Puffer: *16 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Außenliegend*
Besonderheiten: *Einzelblatteinzug optional*





Name: *EPSON LQ-550*
Preis: *DM ca. 1200, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM-Grafik/15 Nationale*
Puffer: *8 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Außenliegend*
Besonderheiten: *Lieferbares Zubehör: Automatisches Einzelblattmagazin, Aufsatzzugtraktor, Zeichensatz-Modul*

Name: *Commodore MPS 1224 C*
Preis: *ca. DM 2000, –*
Typ: *24-Nadel-Drucker*
Zeichensätze: *ASCII/IBM/Internationale*
Puffer: *12 kByte*
Schnittstelle: *Parallel*
Handbuch: *Deutsch*
DIP-Schalter: *Ersetzt durch Tastenfeld*
Besonderheiten: *Fontkarten optional*



Michael Anton

Wenn der Maus-Port nicht mehr will…

Erste Hilfe für eine angeblich defekte Maus

Überraschungen erlebt man auf dem Amiga immer wieder. Diese hier stellt sich als angenehm heraus, da man mit etwas Geschick doch die Inanspruchnahme des Kundendienstes sparen kann…

Das 'Aus für die Maus' war scheinbar über Nacht gekommen: Nach dem Einschalten 'am Tag danach' saß sie nur noch lethargisch in der oberen Bildschirmecke und war durch nichts mehr in Bewegung zu bringen. Erste Diagnose ergab also, daß die Maus 'für die Katz' war. Vor der Konsultation eines Tierarztes wurde jedoch erst noch ein hauseigenes Meßinstrument bemüht – und siehe da, die Maus war von jedweder Lebensenergie abgeschnitten. Dort, wo am Port eigentlich 5 Volt für die Agilität der Maus hätten sorgen sollen, lag nix, oder besser gesagt null an.

Die Maus ist aus – Schnelle Hilfe ist gefragt

Also hatte die Maus bei ihrem Exitus noch versucht, den Rechner mit ins Nirwana zu nehmen? Oder waren die Basteleien, die kurz zuvor noch an diesem Port getestet wurden, doch nicht so ausgereift und hatten das Netzteil ruiniert? Ein Blick in den Rechner sollte für Klarheit sorgen.

Was man so alles erlebt, wenn man die Platine des A2000 in ihrer ganzen Schönheit vor sich sehen will, sei verschwiegen – das silberne Gütesiegel von Commodore zu zertrennen, war jedenfalls noch die einfachste Aufgabe. Nach der Freilegung aller wichtigen Punkte zeigte der erstaunte Blick auf das Meßgerät, daß die 5 Volt, die an den Ports anliegen, am Netzteil noch verfügbar waren. Auch am Störfilter, bestehend aus einem kleinen Kondensator und einer Drossel, die laut Schaltplan die einzigen Hemmnisse im Stromfluß zu den Ports sein sollten, waren sie noch da, danach verlor sich jedoch ihre Spur. Also ein Haarriß auf der Platine? Mitnichten. Ein genauerer Blick auf dieselbe ließ ein weiteres Bauteil im Stromfluß erkennen, welches jedoch nicht im Schaltplan verzeichnet war. Wer nun auf eine Sicherung tippt, der hat recht, wer in diesem Fall auf eine defekte Sicherung tippt, hat noch rechter.

Da es sich bei dieser Sicherung jedoch um eine Lötversion handelt, war der Ersatz nicht so einfach. Die Beschaffung schon, der Einbau jedoch weniger – und schließlich möchte man ja auch weiter mit den Ports experimentieren. Die geniale Lösung bestand nun aus zwei Drähten, die an der defekten Sicherung angelötet und mit einer normalen Steckfassung für Sicherungen verbunden wurden.

Sind Sie unterversichert? Der Amiga nicht – im Gegenteil!

So ist die Lebensversicherung fürs Netzteil gegeben und im Notfall leichter zugänglich, die Maus erfreut sich seither bester Gesundheit. So einfach kann es sein. Nur schade, daß die verantwortliche Sicherung in keinem Schaltplan zu finden war. Ein Schelm, wer Commodore Böses unterstellen möchte. Aber der Befund aus Fakten und Plänen läßt den Verdacht auf einen Riß der Platine zu, der wohl nur durch einen kompletten Austausch zu beheben ist. (Obwohl es auch billiger ginge…)

Ach ja, falls mal ein Zusatz streikt, der seine Versorgung aus der seriellen oder parallelen Schnittstelle bezieht: Auch diese Versorgungsleitungen scheinen gesichert zu sein, wie ein oberflächlicher Blick auf die Platine zeigte. Entsprechende verdächtige Bauteile wurden jedenfalls in der Nähe der Anschlüsse gesichtet. Zur Lokalisierung: Es handelt sich um rötliche Bauteile mit der Form normaler Widerstände. Ihre Beschriftung lautet 'xA', der Aufdruck auf der Platine 'Fx'. Besagter Übeltäter war bei einem A2000 neuerer Bauart das Bauteil F(use)1 mit dem Wert 4A(mpere) und der Leitfähigkeit 0 (kein Durchgang). Zu finden war es direkt neben den Filterbauteilen C218 und FB215 zwischen den Buchsen für Tastatur und Maus (Bild 1), die Sicherungen für die anderen Anschlüsse sind auf dem hinteren Teil der Platine zu finden und durch die Beschriftungen zu erkennen. In jedem Fall muß jedoch zur Reparatur der komplette Laufwerkträger nebst Netzteil demontiert werden. (Vielleicht könnte man ja gleich alle Sicherungen zugänglicher auslagern, beispielsweise in den Schacht für das 5 1/4-Zoll-Floppy: Klappe auf, Sicherung rein, Klappe zu.)

Besitzer älterer A2000 oder kleinerer Geräte müssen sich die entsprechenden Stellen selbst suchen. Die Symptome und Sachverhalte sollten jedoch ähnlich sein.

(jb)



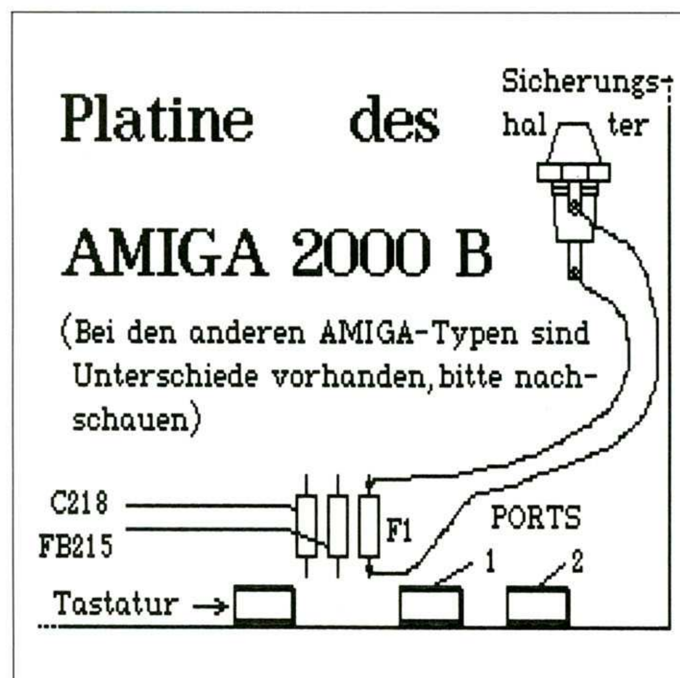


Bild 1: Hier ist beim neueren A2000 die Sicherung versteckt

Wichtig:

Dieser Beitrag soll eine Hilfestellung bei anstehenden Hardwareproblemen sein. Da hier konkret Eigenreparaturen am Amiga beschrieben werden, weisen wir ausdrücklich darauf hin, daß jeder unsachgemäße Eingriff am Rechner mehr Schaden als Nutzen bringen kann. Sollten Sie also nicht zu den 'eingefleischten Lötkolbenrittern' gehören, wenden Sie sich an erfahrene Hardwarebastler.

DER SWIFT 24,

DIE NEUE GENERATION DER FARBIGEN NADELDRUCKER ZU SENSATIONELLEN PREISEN



TESTSIEGER '89 ZU EINEM SAGENHAFTEN PREIS

AMIGA-Test

gut

9,9 von 12	GESAMT-URTEIL AUSGABE10/89

Preis/Leistung	■■■■■
Dokumentation	■■■■
Bedienung	■■■■■■
Verarbeitung	■■■■
Leistung	■■■■■

ZU BEZIEHEN BEI:
MIKY WENNGATZ
JÄGERWEG 31
8031 GILCHING
TEL. 0 81 05/2 45 40
ODER
MW-AUSTRIA
POSTFACH 1 73
A-6700 BLUDENZ
TEL. 0 55 52/6 67 19

CITIZEN

COMPUTER DRUCKER

Es ist so einfach, die vielseitigen Fähigkeiten des Swift24 zu nutzen! Einfach, weil Sie alle Befehle über eine LCD-Anzeige kontrollieren können. Sie brauchen keinen Schraubenzieher und müssen auch nicht nach versteckten DIP-Schaltern suchen – ein einfacher Knopfdruck genügt, um den Swift24 einzustellen.
Neben den üblichen Methoden der Papierzufuhr besitzt der Swift24 die Möglichkeit des Papiereinzuges von unten, einen Schiebe- und Zugtraktor und papiersparendes Abreißen direkt am Seitenende. Sie verlieren also nicht jedesmal eine Leerseite. Die praktische Parkfunktion macht es leicht, von Endlospapier auf Einzelblatteinzug zu wechseln.
Beste Druckqualität ist selbstverständlich. Sie haben die Wahl zwischen Normalschrift und vier Schriften in Korrespondenzqualität. Weiterhin verfügt der Swift24 auch über IBM*, Epson* und NEC* P6 Emulationen.
Bis zu vier gespeicherte Druckformate lassen sich auswählen. Und wenn Sie farbig drucken wollen ist der Swift24 leicht nachrüstbar.
Die Druckgeschwindigkeit beträgt im Normalmodus 192 Zeichen pro Sekunde und bei Schönschrift 64 Zeichen pro Sekunde.
Daß der Swift24 auch die umfassende 2-Jahres-Garantie von Citizen hat, versteht sich ja wohl von selbst.



GUTEN FLUG

MIT AMIGA DOS



Jeder, der zu Hause schon einmal mit einen Flugsimulator gespielt hat, kennt die Problematik des Joystick-Handlings. Der Joystick wirft gerade im Zusammenhang mit Flugsimulatoren spezielle Probleme auf. Nicht nur unlogische Steuerungsumsetzungen plagen den gestreßten Hobbyjetpiloten, es fehlt einfach am Feeling und dem notwendigen Cockpit-Ambiente. Mit ein wenig Zeit, Geduld und der nachfolgend abgedruckten Selbstbauanleitung kann diesem Mißstand nun ein für allemal abgeholfen werden.

Funktionsbeschreibung

Der Flugsimulatorjoystick beruht auf dem digitalen Prinzip, das heißt, wenn man das Steuerhorn nach vorne bewegt, so schließt er den Kontakt wie bei einem handelsüblichen Joystick und nicht wie bei analogen Joysticks Stück für Stück.

Die Funktionen im Überblick:
- Schieben beider Griffe nach vorn:
  -> Flugzeug sinkt
- Ziehen beider Griffe nach hinten:
  -> Flugzeug steigt
- Drehen beider Griffe nach rechts:
  -> Flugzeug dreht rechts
- Drehen beider Griffe nach links:
  -> Flugzeug dreht links
- Drücken des Feuerknopfes:
  -> Waffe wird ausgelöst

Nach diesem allgemeinen Überblick über Funktionsweise und benötigtes Material wollen wir mit dem eigentlichen Bau beginnen. Als Werkzeug sollten Sie zusätzlich, falls vorhanden, eine Heißklebepistole bereitlegen.

Und so wird es gemacht:

1. Von dem 1 m langen Alurohr wird ein zirka 30 cm langes Stück abgeschnitten. An der einen Seite wird eine Einkerbung gefeilt, so daß das andere Rohr genau in diese Kerbung hineinpaßt (senkrecht). Anschließend wird bei der Kerbung der Dübel eingesteckt und durch eine Flamme festgeschmolzen. **Siehe Bild 1**

2. Der Rest des Rohres wird unter Einfluß von Wärme gebogen, so daß es die Form eines "U" erhält. Ein kleiner Tip: Wenn Sie vor dem Biegen das Rohr mit Sand füllen, verwerfen sich die Biegungsstellen nicht so stark, und es entsteht eine gleichmäßige Rundung.
Anschließend wird genau in der Mitte der Konstruktion ein Loch gebohrt und das Rohr in die Einkerbung gesetzt und fest verschraubt.
Siehe Bild 2 und Bild 3

3. Als nächstes werden die Griffe an dem "U" befestigt und die Kabel für die Feuerknöpfe nach unten aus dem Griff herausgeführt.
Siehe Bild 4

4. Im Gehäuse wird die Trennwand mit den Winkeln und den kleinen Schrauben fest mit der Wand des Gehäuses verschraubt. Vorher muß man sich aber vergewissern, daß noch genug Platz für die Platine im vorderen Teil ist.
Siehe Bild 5

5) Anschließend wird von vorn (nicht von der Seite,

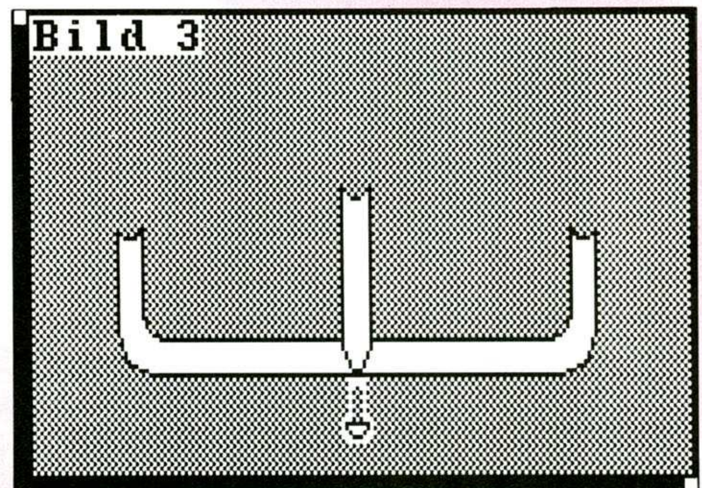


Bild 3

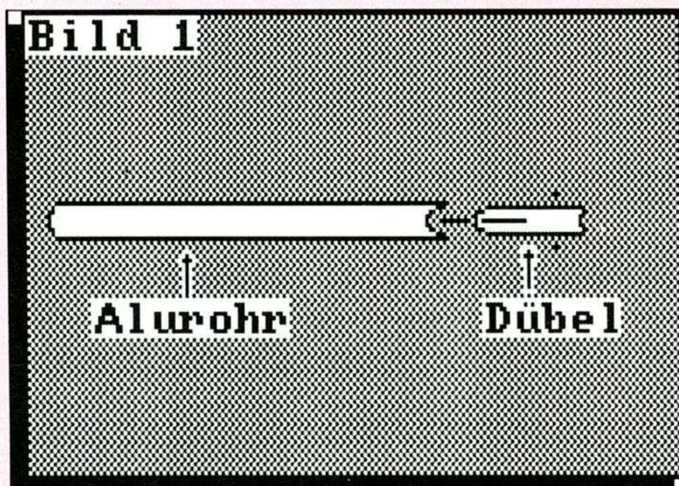


Bild 1

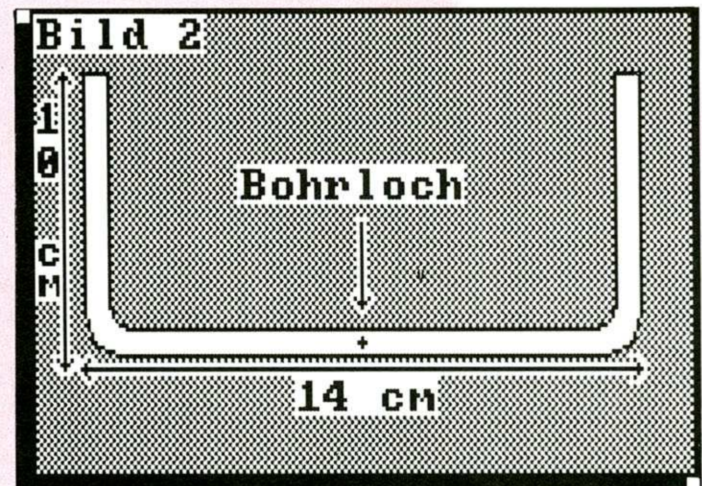


Bild 2

wo die Platine untergebracht ist) ein Loch durch das Gehäuse und durch die Trennwand gebohrt und die Stange mit den beiden Joystickgriffen durchgeführt.
Siehe Bild 6

Im Zwischenraum werden nun die beiden übrigen Schrauben in einer Entfernung von zirka 8 mm von Trennwand und Gehäuse eingeschraubt, so daß der Griff nicht mehr aus dem Gehäuse herausrutschen kann.

6. Jetzt wird genau in der Mitte zwischen den beiden Schrauben die große Schrau-

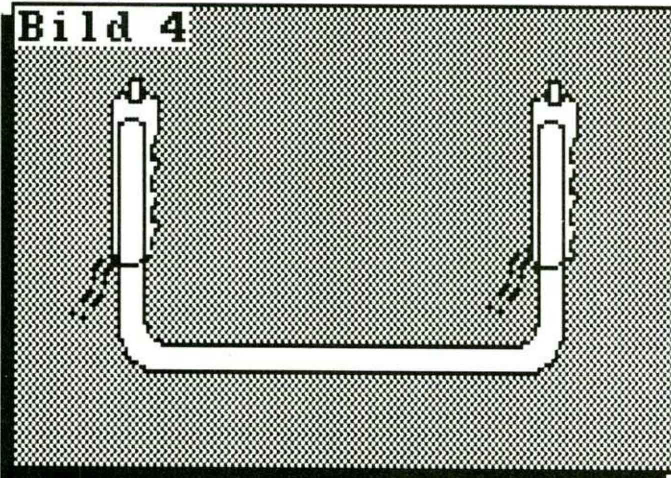


Bild 4

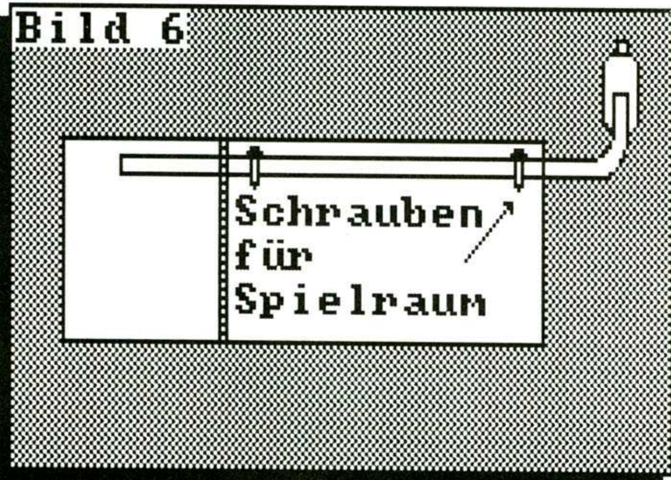


Bild 6

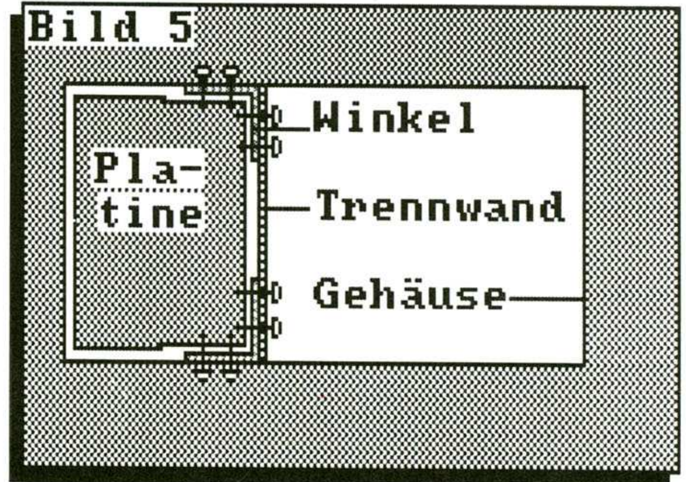


Bild 5

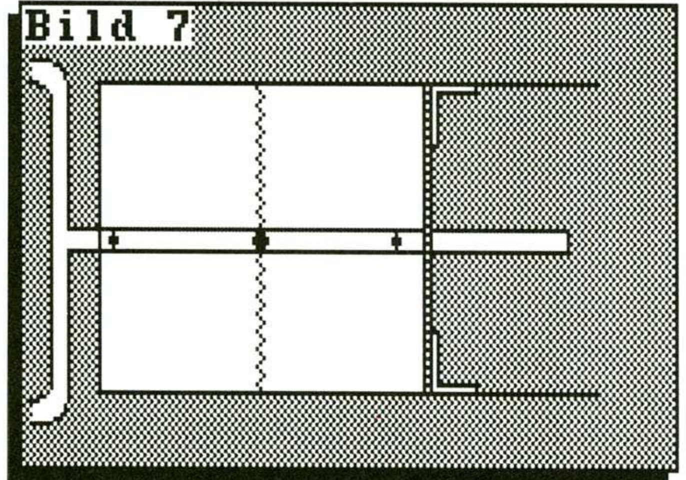


Bild 7

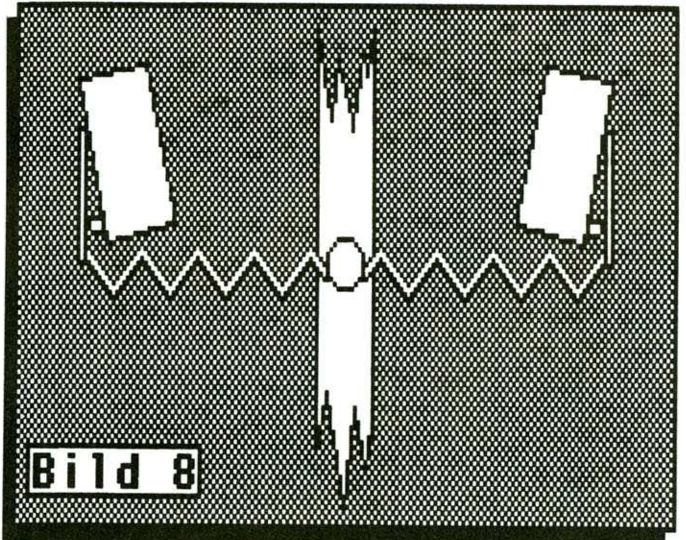


Bild 8

AMIGA-DOS-CHECKLISTE
Stückliste für den Flugsimulatorjoystick

- ○ 4x Mikroschalter mit Federarm
- ○ ca. 1 m Alurohr mit 0,75 cm Durchmesser
- ○ 2 Griffe von einem alten Joystick, (am besten Quickshot II)
- ○ 1 noch funktionsfähige Platine von einem Joystick (mit oder ohne Autofeuer)
- ○ 10 kleine Schrauben mit Muttern
- ○ 1 große Schraube mit Mutter
- ○ 2 starke Federn
- ○ 2 sehr schwache Federn, die ausreichen, um den Mikroschalter auszulösen
- ○ 1 Gehäuse (L/B/H 20cm/11cm/5cm)
- ○ 4 Saugfüße
- ○ 2 Winkel (klein)
- ○ 1 Trennwand, die genau ins Gehäuse paßt
- ○ 1 Dübel, der in das Alurohr hineinpaßt
- ○ 1 Schraube, passend zum Dübel

be eingesetzt und festgezogen. Der Joystick wird jetzt so eingestellt, daß nach beiden Seiten der gleiche Spielraum zur Verfügung steht. Danach werden genau parallel zu der großen Schraube in der Mitte die starken Federn zu beiden Seiten des Gehäuses gespannt, dadurch wird eine Nullstellung aus allen Lagen und Stellungen des Joystickgriffes bewirkt. Wenn Sie also das Steuerhorn loslassen, kehrt der Joystick automatisch in die Nullstellung zurück. **Siehe Bild 7**

Achtung: Die Federn müssen am untersten Ende der großen Schraube befestigt werden, sonst reicht die Federkraft nicht aus, um die Nullstellung hervorzurufen.

7. In die Federn werden nun die Spannarme der Mikroschalter eingesetzt und mit Klebstoff, am besten Heißkleber, befestigt.
Siehe Bild 8

8. Nun werden die schwachen Federn mit dem Spannarm verbunden und oberhalb der starken Federn an der großen Schraube befestigt. Die Schalter werden unterhalb der Aluminiumstange vor und nach der großen Schraube befestigt und angeschlossen.
Siehe Bild 9

9. Nun müssen nur noch alle Kontakte mit der Platine verbunden, das Gehäuse verschraubt und die Feuerknöpfe angeschlossen werden.
Wer eine Platine mit Autofeuer zur Verfügung hat, kann den Schalter nach außen setzen und somit noch eine tolle Autofeuerfunktion erhalten.
Mit diesem Joystick lassen sich eigentlich alle Simulationsspiele wie Interceptor F-18, Falcon F-16, aber auch Spiele wie Test Drive und Hybris hervorragend spielen. Der Joystick erzeugt ein echtes Fluggefühl, und bei einem Autorennsimulator kommt es dem Spieler manchmal vor, als würde er tatsächlich in diesem Fahrzeug sitzen. Viel Spaß beim Fliegen wünscht Ihnen Ihre AMIGA-DOS-Redaktion!

Und denken Sie daran: Es ist schon mancher Meister vom Himmel gefallen...

(mm)

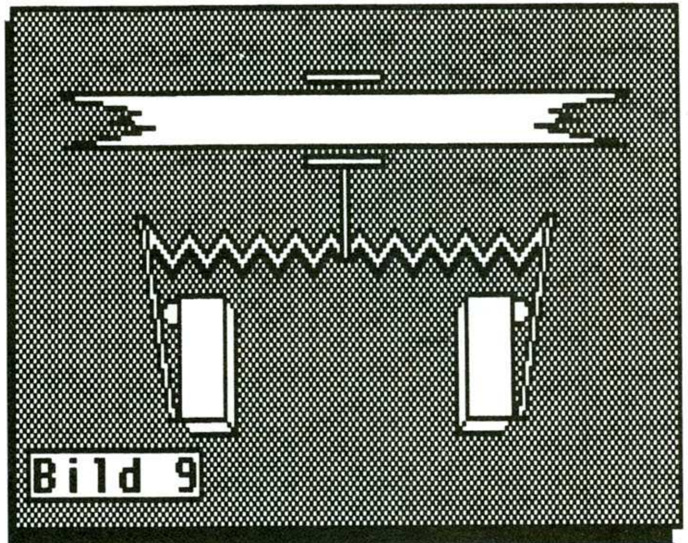
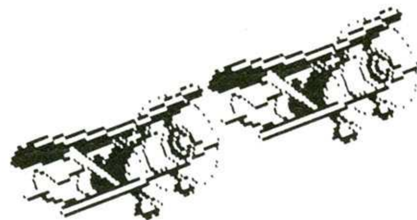


Bild 9



SPIELEN SIE GERNE?

Lieben Sie es, in finsteren und unheimlichen Gewölben den letzten Geheimnissen auf den Grund zu gehen? Zeichnen Sie gerne Karten von Gewölben und Spiellandschaften?

Dann sind Sie der richtige Partner für uns.

Denn für unsere AMIGA-DOS-Spieletips suchen wir ständig Tips, Tricks, Karten, Cheatmodes und alles andere, was das Spielerherz höher schlagen läßt. Wenn Sie interessante Informationen, Karten oder Lösungsvorschläge für Computerspiele herausgefunden haben, sind Sie unser(e) Mann/Frau.

Durch Ihre Information können Sie vielen Spielern weiterhelfen, die ohne Ihre Hilfe unter Umständen monatelang an der gleichen Problematik "festhängen". Oft ist es nur ein Wort oder ein kleiner Hinweis, der dem Spielgeschehen neues Leben einhaucht.

Aus diesen Gründen möchten wir Sie um Ihre Mitarbeit bitten.

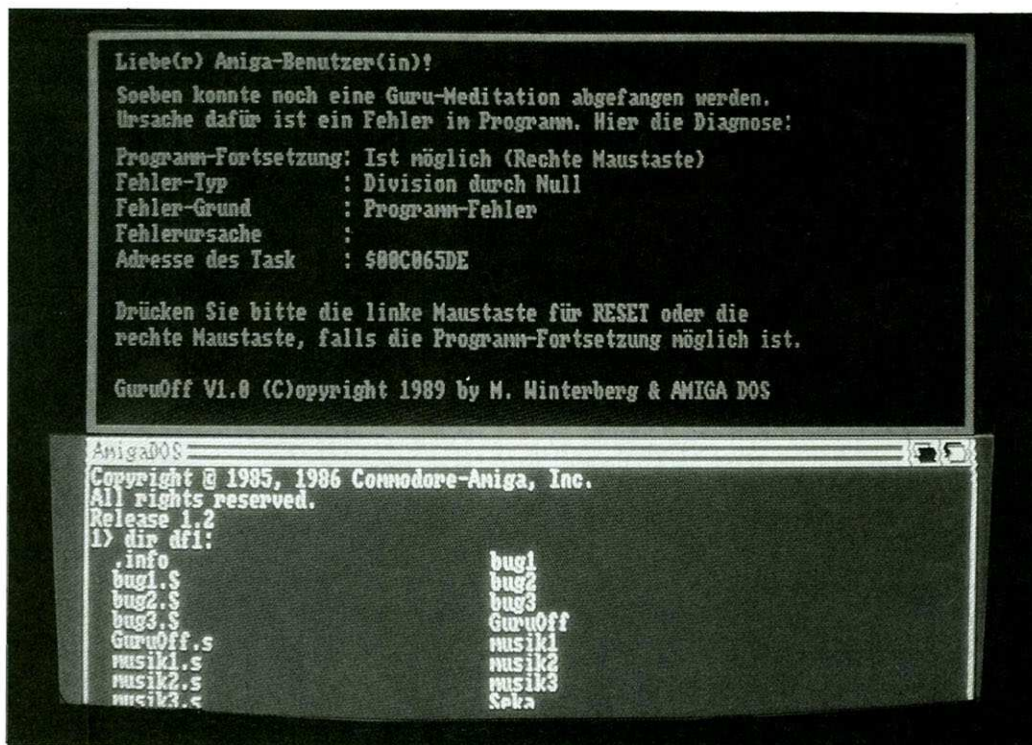
Wenn Sie also über Spieletips – gleich welcher Art – verfügen, würden wir uns freuen, von Ihnen Post zu bekommen.

Und hier nun die Adresse, an die Sie Ihre Spieletips schicken können:

DMV Verlag
Redaktion AMIGA DOS
Stichwort: Spieletips
Postfach 250
3440 Eschwege

PS: Unter allen Einsendungen verlosen wir jeden Monat fünf Softwaretitel für die besten Spieletips.





Mit GuruOff steht ein nützliches Werkzeug ins Haus, das nicht nur für den Programmierer, der seine Programme austesten will, geeignet ist

Michael Winterberg

GuruOff – Dem Guru ein Schnippchen geschlagen

Jetzt ist Schluß mit unübersichtlichen Meditationen

Welcher Amiga-Besitzer kennt nicht das folgende Problem? Da sitzt man hinter seiner Kiste, hat stundenlang etwas eingetippt, und plötzlich schlägt der Guru mit einer Meditation zu. Die ganze Arbeit für die Katz? Das muß nicht sein, denn unser Programm GuruOff hat die Aufgabe, Gurus abzufangen.

Nachdem ich in der Edition 68000 unserer Schwesterzeitschrift toolbox den Artikel "Decodierung des Guru-Requesters" gelesen hatte, kam mir der Gedanke, dieses Thema auszubauen. Zwar erkannte ich durch den Artikel schnell den Grund für die Gurus, doch war mir das ewige Heft-Blättern im Falle einer Meditation doch zu nervig. So beschloß ich, ein Assembler-Programm zu schreiben, das die Guru-Meldung auswertet.

Ich fand sehr schnell heraus, daß die Alert-Funktion in der Exec-Library (Offset -108) für die Guru-Meldung zuständig ist.

Daraufhin schrieb ich mir eine eigene Requester-, sowie eine Initialisierungs-Routine, die die Einsprungadresse vom Alert auf meine Routine "umbog". Der "einfache Guru" war somit geboren... Dieser hatte jedoch noch den Fehler, daß er die Exceptions, also den Ausnahmezustand, nicht abfangen konnte. Eine Division durch Null brachte also immer noch die ursprüngliche Guru-Meldung.

Für die Exceptions habe ich dann die Routinen geschrieben, die später meinen Guru-Requester anzeigen sollen.

Es ist jedoch leider nicht möglich, sämtliche Exceptions abzufangen, da einige Programme beispielsweise die Illegal Instruction-Exception benutzen.

Wir holen den Guru von seiner Wolke

In vielen Fällen ist es aber möglich, das "abgestürzte" Programm weiterlaufen zu lassen. Ob dies möglich ist, wird in der neuen Guru-Meldung dann angezeigt.

Das Prinzip der neuen Guru-Routine ist eigentlich ganz einfach: Sie bekommt die Daten des ursprünglichen Gurus übermittelt.

Aus den Tabellen im Programm, das übrigens mit dem Seka-Assembler geschrieben wurde, wird dann der Ausgabetext für die Fehlerursache ermittelt und angezeigt.

(br)

Listings

```
; Guru-Abfangroutine
;
; Autor: Michael Winterberg
;
; (C)opyright 1989 by Michael Winterberg & AMIGA DOS

        move.l  4,a6                    ; Exec-Basis nach A6
        suba.l  a1,a1                   ; Eigenen Task er-
                                        ; mitteln
        jsr     -294(a6)                ; Task suchen
        move.l  d0,a4                   ; Task nach A4
        tst.l   172(a4)                 ; Start von Work-
                                        ; bench aus?
        bne.s   noworkbench             ; Nein, dann ver-
                                        ; zweige
        lea     92(a4),a0               ; Rastport nach A0
        jsr     -384(a6)                ; Auf Nachricht
                                        ; warten
        lea     92(a4),a0               ; Rastport nach A0
        jsr     -372(a6)                ; Nachricht abholen
        bra     run                     ; Zum Hauptprogramm
dos:                                    ; Label für Dos-
        dc.b    "dos.library",0         ; Library-Text
dosbase:                                ; Label für Zeiger
        dc.l    0                       ; auf Dos-Library
clitext:                                ; Auszugebener Text
        dc.b    10                      ; wenn von CLI ge-
                                        ; startet
        dc.b    "GuruOff V1.0 (C)opyright 1989 by"
        dc.b    " Michael Winterberg & AMIGA DOS"
        dc.b    10,"Installiert...",10,10
even
noworkbench:                            ; CLI-Start
        lea     dos(pc),a1              ; Textzeiger auf
                                        ; "Dos.Library" nach
                                        ; A1
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -408(a6)                ; Library öffnen
        move.l  d0,dosbase              ; DosBase sichern
        move.l  d0,a6                   ; DosBase nach A6
        jsr     -60(a6)                 ; Output-Handle er-
                                        ; mitteln
        move.l  d0,d1                   ; Output-Handle nach
                                        ; D1
        move.l  #clitext,d2             ; Zeiger auf Text
                                        ; Nach D2
        move.l  #80,d3                  ; Anzahl nach D3
        move.l  dosbase(pc),a6          ; DosBase nach A6
        jsr     -48(a6)                 ; WRITE, Text aus-
                                        ; geben
        move.l  dosbase(pc),a1          ; DosBase nach A1
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -414(a6)                ; Library schließen
run:                                    ; Hauptprogramm
        bsr     copy                    ; Guru-Abfang-Pro-
                                        ; gramm an freien
                                        ; Speicher kopieren
        move.l  #execpt2,d0             ; Berechnen der
        sub.l   #start,d0               ; neuen Adressen für
        add.l   speicher(pc),d0         ; Execption 2,3
        move.l  d0,$008                 ; und 5
        move.l  #execpt3,d0
        sub.l   #start,d0
        add.l   speicher(pc),d0
        move.l  d0,$00c
```

```
        move.l  #execpt5,d0
        sub.l   #start,d0
        add.l   speicher(pc),d0
        move.l  d0,$014
        move.l  #$60a,a1          ; Zeiger auf DISPLAY
                                  ; ALERT nach A1
        move.l  code+2(pc),d0     ; Einsprungadresse
                                  ; nach D0
        sub.l   #start,d0         ; Neu berechnen
        add.l   speicher(pc),d0
        move.l  d0,code+2         ; und sichern
        lea     code(pc),a0       ; Zeiger auf Ein-
                                  ; sprung-JMP nach A0
        move.l  (a0)+,(a1)+       ; und nach DISPLAY
        move.w  (a0)+,(a1)+       ; ALERT kopieren
        clr.l   d0                ; D0 löschen
        rts                       ; Fertig
start:                            ; Start-Label, ab
                                  ; hier wird das Pro-
                                  ; gramm gesichert
execpt2:                          ; Exception 2
        move.l  #$80000002,d7     ; Guru-Nr. nach D7
        move.l  #0,a5             ; Task-Adresse = 0
        bsr     showguru          ; Guru anzeigen
        rte                       ; Fertig
execpt3:                          ; Exception 3
        move.l  #$80000003,d7     ; Guru-Nr. nach D7
        move.l  #0,a5             ; Task-Adresse = 0
        bsr     showguru          ; Guru anzeigen
        rte                       ; Fertig
execpt5:                          ; Exception 5
        move.l  #5,d7             ; Guru-Nr. nach D7
        move.l  2(sp),a5          ; Rücksprung-Adresse
                                  ; Nach A5
        bsr     showguru          ; Guru anzeigen
        rte                       ; Fertig
code:                             ; Einsprung-Kommando
        bra     showguru          ; Verzweige zu "Guru
                                  ; anzeigen"
fxx:                              ; Text für Fehler-
                                  ; Meldung
        dc.b    0
        dc.b    $00,$fe,"Absturz",0
        dc.b    $00,$01,"Exec.Library",0
        dc.b    $00,$02,"Graphics.Library",0
        dc.b    $00,$03,"Layers.Library",0
        dc.b    $00,$04,"Intuition.Library",0
        dc.b    $00,$05,"Math.Library",0
        dc.b    $00,$06,"Clist.Library",0
        dc.b    $00,$07,"Dos.Library",0
        dc.b    $00,$08,"Ram.Library",0
        dc.b    $00,$09,"Icon.Library",0
        dc.b    $00,$0a,"Expansion.Library",0
        dc.b    $00,$10,"Audio.Device",0
        dc.b    $00,$11,"Console.Device",0
        dc.b    $00,$12,"GamePort.Device",0
        dc.b    $00,$13,"KeyBoard.Device",0
        dc.b    $00,$14,"Track-Disk.Device",0
        dc.b    $00,$15,"Timer.Device",0
        dc.b    $00,$20,"CIA-Fehler",0
        dc.b    $00,$21,"Disk-Fehler",0
        dc.b    $00,$22,"Misc-Fehler",0
        dc.b    $00,$30,"BootStrap-Fehler",0
        dc.b    $00,$31,"Workbench-Fehler",0
        dc.b    $00,$32,"DiskCopy-Fehler",0
        dc.b    $ff,$ff
fxx2:                             ; Text für Fehler-
                                  ; Meldung
        dc.b    0
        dc.b    $00,$02,"Busfehler",0
        dc.b    $00,$03,"Adressierungsfehler",0
        dc.b    $00,$04,"Illegale Instruktion",0
        dc.b    $00,$05,"Division durch Null",0
        dc.b    $00,$06,"CHR Instruktion",0
        dc.b    $00,$07,"TRAPV Instruktion",0
        dc.b    $00,$08,"Privileg-Verletzung",0
        dc.b    $00,$09,"Trace-Modus",0
        dc.b    $00,$0a,"1010 Op-Code",0
        dc.b    $00,$0b,"1111 Op-Code",0
        dc.b    $ff,$ff
even
fyy:                              ; Text für Fehler-
                                  ; Meldung
        dc.b    0
        dc.b    $00,$fe,"Programm-Fehler",0
        dc.b    $00,$01,"Zu wenig RAM",0
        dc.b    $00,$02,"Library erzeugen",0
        dc.b    $00,$03,"Library öffnen (Library nicht"
        dc.b    " verfügbar)",0
        dc.b    $00,$04,"Device öffnen (Device nicht"
        dc.b    " verfügbar)",0
        dc.b    $00,$05,"Öffnen der Resources (nicht"
        dc.b    " verfügbar)",0
        dc.b    $00,$06,"I/O-Signal",0
        dc.b    $00,$07,"Kein Signal",0
even
fehler:                           ; Text für Fehler-
                                  ; Meldung
        dc.b    0
        dc.b    $01,$01,"Prüfsumme Exception-Vektor",0
        dc.b    $01,$02,"Prüfsumme Exec-Base",0
        dc.b    $01,$03,"Prüfsumme Library",0
        dc.b    $01,$04,"Kein Speicher frei für "
        dc.b    "Library",0
        dc.b    $01,$05,"Daten-Fehler in Speicherliste"
        dc.b    0
        dc.b    $01,$06,"Speicher für Interrupt fehlt"
        dc.b    0
        dc.b    $01,$07,"InitAPtr-Fehler",0
        dc.b    $01,$08,"Fehlerhafte Semaphore",0
        dc.b    $01,$09,"Speicher doppelt freigegeben"
        dc.b    0
        dc.b    $01,$0a,"Exception-Vektor fehlerhaft",0
        dc.b    $02,$01,"RAM für Copperliste fehlt",0
        dc.b    $02,$02,"RAM für CopperInstruct-Liste "
        dc.b    "fehlt",0
        dc.b    $02,$03,"Überladene Copperliste",0
        dc.b    $02,$04,"Überladene Copperliste",0
        dc.b    $02,$05,"RAM für Copperliste fehlt",0
        dc.b    $02,$06,"RAM für Long Frame fehlt",0
        dc.b    $02,$07,"RAM für Short Frame fehlt",0
        dc.b    $02,$08,"RAM für Flächenfüllen fehlt"
        dc.b    0
        dc.b    $02,$09,"Kein RAM für Textausgabe",0
        dc.b    $02,$0a,"RAM für BlitterBitMap fehlt",0
        dc.b    $02,$0b,"Fehlerhafte Speicherbereichs"
        dc.b    "angabe",0
        dc.b    $02,$30,"MakeVPort-Fehler",0
        dc.b    $14,$34,"RAM für Notfallspeicher fehlt"
        dc.b    0
        dc.b    $03,$01,"RAM für Layer fehlt",0
        dc.b    $04,$01,"Gadget-Typ unbekannt",0
        dc.b    $04,$02,"RAM für neuen Port fehlt",0
        dc.b    $04,$03,"RAM für Anzeigen einer Menü"
        dc.b    "zeile fehlt",0
        dc.b    $04,$04,"RAM für Anzeigen des Submenüs"
        dc.b    " fehlt",0
        dc.b    $04,$05,"RAM für Anzeigen der Menükopf"
        dc.b    "zeile fehlt",0
        dc.b    $04,$06,"Fehlerhafte Menü-Kopfzeilen-"
        dc.b    "Position",0
        dc.b    $04,$07,"RAM für Öffnen eines Screens "
        dc.b    "fehlt",0
        dc.b    $04,$08,"RAM für einen RastPort fehlt"
        dc.b    0
        dc.b    $04,$09,"Falscher Screen-Typ",0
        dc.b    $04,$0a,"RAM für Gadget fehlt",0
        dc.b    $04,$0b,"RAM für Window fehlt";0
        dc.b    $04,$0c,"Intuition-Systemstatus nicht "
        dc.b    "korrekt",0
        dc.b    $04,$0d,"IDCMP-Message-Form ist fehler"
        dc.b    "haft",0
        dc.b    $04,$0e,"Message-Inhalt ist nicht zu "
        dc.b    "verarbeiten",0
        dc.b    $04,$0f,"RAM für Console-Device fehlt"
        dc.b    0
        dc.b    $05,$00,"Unbekannter Fehler in Math-"
        dc.b    "Library",0
        dc.b    $06,$00,"Unbekannter Fehler in Clist-"
        dc.b    "Library",0
        dc.b    $07,$01,"RAM fehlt nach dem Booten",0
        dc.b    $07,$02,"Fehler bei Task beenden",0
        dc.b    $07,$03,"QPunkt-Fehler",0
        dc.b    $07,$04,"Empfangendes Datenpaket nicht"
        dc.b    " zugelassen",0
        dc.b    $07,$05,"Free-Vector Funktions-Fehler"
        dc.b    0
        dc.b    $07,$06,"Fehlerhafter Block auf Disk",0
        dc.b    $07,$07,"Fehlerhafte/zerstörte Bitmap "
        dc.b    "auf Disk",0
        dc.b    $07,$08,"Bereits gelöschter Key",0
        dc.b    $07,$09,"Fehlerhafte Block-Prüfsumme",0
        dc.b    $07,$0a,"Schreib/-Lesefehler auf Disk"
        dc.b    0
        dc.b    $07,$0b,"Fehlerhafte Key-Nummer",0
        dc.b    $07,$0c,"Nicht zugelassenes Overlay",0
        dc.b    $08,$01,"Fehlerhafter Eintrag in RAM-"
        dc.b    "Segmentliste",0
        dc.b    $09,$00,"Siehe Fehler-Grund",0
        dc.b    $0a,$00,"Siehe Fehler-Grund",0
        dc.b    $10,$00,"Siehe Fehler-Grund",0
        dc.b    $11,$00,"Siehe Fehler-Grund",0
        dc.b    $12,$00,"Siehe Fehler-Grund",0
        dc.b    $13,$00,"Siehe Fehler-Grund",0
        dc.b    $14,$00,"Siehe Fehler-Grund",0
        dc.b    $0a,$01,"Such-Fehler auf Disk",0
        dc.b    $0a,$02,"Fehler beim Warten auf einen "
        dc.b    "Timerimpuls",0
        dc.b    $15,$01,"Fehlerhafter Zugriff auf "
        dc.b    "Timer-Device",0
        dc.b    $15,$02,"Fehlerhafte/nicht vorhandene "
        dc.b    "Netzfrequenz",0
        dc.b    $20,$00,"Siehe Fehler-Grund",0
        dc.b    $21,$01,"Diskette bereits im Laufwerk"
        dc.b    0
        dc.b    $21,$02,"Laufwerk nicht aktiv",0
        dc.b    $22,$00,"Siehe Fehler-Grund",0
        dc.b    $30,$01,"Fehler beim Booten",0
        dc.b    $31,$0c,"Icon-Library fehlt",0
        dc.b    $31,$00,"Siehe Fehler-Grund xx",0
        dc.b    $32,$09,"Icon-Library fehlt",0
        dc.b    $32,$00,"Siehe Fehler-Grund",0
        dc.b    $ff,$ff
text:                             ; Text für Bild-
                                  ; schirmausgabe
        dc.b    0,20,15
        dc.b    "Liebe(r) Amiga-Benutzer(in)!",0
        dc.b    10,10,20,25
        dc.b    "Soeben konnte noch eine Guru-"
        dc.b    "Meditation abgefangen "
        dc.b    "werden.",0,10,10,20,35
        dc.b    "Ursache dafür ist ein Fehler im "
        dc.b    "Programm. Hier die "
        dc.b    "Diagnose:",0,10,10,20,50
        dc.b    "Programm-Fortsetzung: "
m1:                               ; Text-Feld 1
```

Guru-Off

```
                blk.b   50,$20                  ; Text-Buffer
                dc.b    0,10,10,20,60
                dc.b    "Fehler-Typ               : "
        m2:                                     ; Text-Feld 2
                blk.b   50,$20                  ; Text-Buffer
                dc.b    0,10,10,20,70
                dc.b    "Fehler-Grund             : "
        m3:                                     ; Text-Feld 3
                blk.b   50,$20                  ; Text-Buffer
                dc.b    0,10,10,20,80
                dc.b    "Fehlerursache            : "
        m4:                                     ; Text-Feld 4
                blk.b   50,$20                  ; Text-Buffer
                dc.b    0,10,10,20,90
                dc.b    "Adresse des Task         : "
        m5:                                     ; Text-Feld 5
                blk.b   50,$20                  ; Text-Buffer
                dc.b    0,10,10,20,110
                dc.b    "Drücken Sie bitte die linke Maustaste"
                dc.b    " für RESET oder"
                dc.b    " die"
                dc.b    0,10,10,20,120
                dc.b    "rechte Maustaste, falls die Programm-"
                dc.b    "Fortsetzung möglich ist."
                dc.b    0,10,10,20,140
                dc.b    "GuruOff V1.0 (C)opyright 1989 by "
                dc.b    "M. Winterberg & AMIGA DOS",0
                dc.b    0,0,0,0
        lib:                                    ; Text für
                                                ; Intuition-Library
                dc.b    "intuition.library",0
        even
        libbase:                                ; Label für
                                                ; Intuition-Basis-
                dc.l    0                       ; Adresse
        putdata:                                ; Daten in Feld ein-
                                                ; fügen
                clr.l   d0                      ; Zähler löschen
        putdataloop:                            ; Schleife
                move.b  (a0,d0),(a1,d0)         ; 1 Zeichen kopieren
                tst.b   (a0,d0)                 ; Ende?
                beq.s   putdataweiter           ; Wenn ja, verzweige
                addq.l  #1,d0                   ; Zähler +1
                cmp.l   #47,d0                  ; 47 Bytes erreicht?
                bne.s   putdataloop             ; Wenn nein, verzw.
        putdataweiter:                          ; Rest auffüllen
                move.b  #$20,(a1,d0)            ; Leerzeichen
                                                ; kopieren
                addq.l  #1,d0                   ; Zähler +1
                cmp.l   #50,d0                  ; 50 erreicht?
                bne.s   putdataweiter           ; Wenn nein, verzw.
                rts                             ; Fertig
        suchetext:                              ; Text suchen
                tst.b   (a0)                    ; Endemarkierung ge-
                                                ; funden?
                beq.s   suchetextok             ; Wenn ja, verzweige
                addq.l  #1,a0                   ; Adresse +1
                bra.s   suchetext               ; Zur Schleife
        suchetextok:                            ; weiter
                cmp.b   #$ff,1(a0)              ; Tabellen-Ende?
                beq.s   suchetextende           ; Wenn ja, verzweige
                tst.b   2(a0)                   ; Fehlt Eintrag?
                beq.s   suchetext2              ; Wenn ja, verzweige
                clr.l   d1                      ; D1 löschen
                move.b  1(a0),d1                ; Tabelle-Nr nach D1
                asl.l   #8,d1                   ; Um 8 Bits nach
                                                ; links verschieben
                add.b   2(a0),d1                ; Tabellen-Pos dazu
                                                ; addieren
                cmp.l   d0,d1                   ; Stimmt mit der
                                                ; Such-Nr überein?
                beq.s   suchetextfertig         ; Wenn ja, verzweige
                addq.l  #3,a0                   ; Adresse +3
                bra.s   suchetext               ; Zur Schleife
        suchetext2:                             ; Kein Eintrag
                move.l  d0,d1                   ; Such-Nr nach D1
                and.l   #$ff00,d1               ; Such-Tabelle-Nr
                                                ; sichern
                clr.l   d2                      ; D2 löschen
                move.b  1(a0),d2                ; Tabelle-Nr nach D2
                asl.l   #8,d2                   ; Nach links um 8
                                                ; Bit verschieben
                cmp.l   d1,d2                   ; Such-Tabelle?
                beq.s   suchetextfertig         ; Wenn ja, verzweige
                addq.l  #3,a0                   ; Adresse +3
                bra.s   suchetext               ; Zur Schleife
        suchetextfertig:                        ; Zeiger auf Text
                                                ; setzen
                addq.l  #3,a0                   ; Adresse +3
                rts                             ; Fertig
        suchetextende:                          ; Nicht gefunden
                lea     leer(pc),a0             ; Zeiger auf Leer-
                                                ; Buffer
                rts                             ; Fertig
        leer:                                   ; Leerbuffer
```

Haben Sie Fragen zum Thema AMIGA und AMIGA DOS?

Dann rufen Sie doch einfach mal an!

Das AMIGA-DOS-Team freut sich darauf, Ihnen bei Fragen zum AMIGA und zu diesem Magazin Hilfestellung zu leisten. In diesem Sinne steht Ihnen das AMIGA-DOS-Team an jedem **Dienstag** in der Zeit von **17:00 bis 20:00 Uhr** zur Verfügung, um Ihre Fragen zu beantworten. Wählen Sie einfach eine der nachfolgenden Nummern, um direkt den betreffenden Redakteur zu erreichen:

Jürgen Borngießer:
05651/809-742

Bernhard Rinke:
05651/809-743

Heinrich Stiller:
05651/809-741

Markus Matejka:
05651/809-740

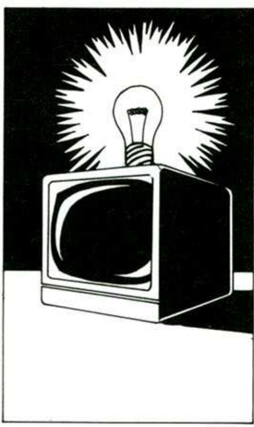
Wir freuen uns auf Ihren Anruf!

Ihr AMIGA-DOS-TEAM



```
        dc.b    " ",0
m1a:
        dc.b    "Ist möglich (Rechte Maustaste)",0
m1b:
        dc.b    "Ist nicht möglich!",0
m1c:
        dc.b    "Ist nicht möglich, Speichermangel!",0
even
hextext:
        dc.b    "$00000000",0
hexdata:
        dc.b    "0123456789ABCDEF"
even
gethex:                                 ; Dezimal in Hexa-
                                        ; dezimal umwandeln
        move.l  a5,d1                   ; Dez-Zahl nach D1
        lea     hextext+9(pc),a0        ; Zeiger auf Hex-
                                        ; Text nach A0
        lea     hexdata(pc),a1          ; Zeiger auf Hex-
                                        ; Daten nach A1
        clr.l   d0                      ; Zähler löschen
gethexloop:                             ; Schleife
        move.l  d1,d2                   ; Dez-Zahl nach D2
        and.l   #$f,d2                  ; 1 Hex-Ziffer
                                        ; sichern
        move.b  (a1,d2),-(a0)           ; Hex-Ziffer als
                                        ; Text kopieren
        asr.l   #4,d1                   ; Nächste Ziffer
        addq.l  #1,d0                   ; Zähler +1
        cmp.l   #8,d0                   ; 8 Ziffern?
        bne.s   gethexloop              ; Wenn nein, verzw.
        rts                             ; Fertig
showguru:                               ; Fehler anzeigen
        movem.l d0-d7/a0-a6,-(a7)       ; Register auf Stack
        move.l  d7,d6                   ; Guru-Nr. nach D6
        lea     m1a(pc),a0              ; Textzeiger nach A0
        lea     m1(pc),a1               ; Zeiger auf Text-
                                        ; feld nach A1
        and.l   #$80000000,d6           ; Guru-Art sichern
        tst.l   d6                      ; = 0?
                                        ; (0 = Programm kann
                                        ;      weiterlaufen,
                                        ;  8 = Programm kann
                                        ;      nicht weiter-
                                        ;      laufen)
        beq.s   showguru1               ; Wenn ja, verzweige
        lea     m1b(pc),a0              ; Textzeiger nach A0
showguru1:                              ; weiter
        bsr     putdata                 ; Daten einfügen
        move.l  d7,d6                   ; Guru-Nr nach D6
        lea     fxx2(pc),a0             ; Fehler-Tabelle
                                        ; nach A0
        and.l   #$7fffffff,d6           ; Library bzw Excep-
                                        ; tion ermitteln
        cmp.l   #15,d6                  ; Ist Exception?
        bmi.s   showguru2               ; Wenn ja, verzweige
        and.l   #$7f000000,d6           ; Library ermitteln
        asr.l   #8,d6
        asr.l   #8,d6
        asr.l   #8,d6
        lea     fxx(pc),a0              ; Fehler-Tabelle
                                        ; nach A0
        tst.l   d6                      ; Library?
        bne.s   showguru2               ; Wenn ja, verzweige
        move.l  #$fe,d6
showguru2:
        move.l  d6,d0
        bsr     suchetext               ; Text raussuchen
        lea     m2(pc),a1               ; Feld nach A1
        bsr     putdata                 ; Daten einfügen
        move.l  d7,d6                   ; Guru-Nr nach D6
        and.l   #$ffffff,d6             ; Absturzgrund
        asr.l   #8,d6                   ; ermitteln
        asr.l   #8,d6
        move.l  d6,d0
        move.l  d6,d5
        cmp.l   #0,d0
        bne     showguru3
        move.l  #$fe,d0
showguru3:
        lea     fyy(pc),a0              ; Fehler-Tabelle
                                        ; nach A0
        bsr     suchetext               ; Text raussuchen
        lea     m3(pc),a1               ; Feld nach A1
        bsr     putdata                 ; Daten einfügen
        move.l  d7,d6                   ; Guru-Nr nach A1
        and.l   #$7ffffff,d6            ; Absturz-Ursache
        asr.l   #8,d6                   ; ermitteln
        asr.l   #8,d6
        move.l  d7,d5
        and.l   #$1fff,d5
        add.l   d5,d6
        move.l  d6,$0110
        move.l  d6,d0
        lea     fehler(pc),a0           ; Fehler-Tabelle
                                        ; nach A0
        bsr     suchetext               ; Text raussuchen
        lea     m4(pc),a1               ; Feld nach A1
        bsr     putdata                 ; Daten einfügen
        bsr     gethex                  ; Task-Adresse (A5)
                                        ; ermitteln
        lea     hextext(pc),a0          ; Adresse-Text nach
                                        ; A0
        lea     m5(pc),a1               ; Feld nach A1
        bsr     putdata                 ; Daten einfügen
        move.l  4,a6                    ; ExecBase nach A6
        moveq   #2,d1                   ; SpeicherTyp =
                                        ; ChipRam
        jsr     -216(a6)                ; Wieviel ist ver-
                                        ; fügbar?
        cmp.l   #25000,d0               ; Wenn kleiner als
        bmi.s   reset                   ; 25000 dann ver-
                                        ; zweige
        cmp.l   #1,d5                   ; Fehler = Zu wenig
                                        ; RAM?
        beq.s   reset                   ; Wenn ja, verzweige
        lea     lib(pc),a1              ; Zeiger auf Intui-
                                        ; tion-Name nach A1
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -408(a6)                ; Library öffnen
        lea     libbase(pc),a0          ; Base sichern
        move.l  d0,(a0)
        lea     text(pc),a0             ; Zeiger auf Fehler-
                                        ; Text
        move.l  #$a0,d1                 ; Höhe in Bild-
                                        ; punkten
        move.l  #$30000000,d0           ; Flag setzen
        move.l  libbase(pc),a6          ; Intuition-Basis
                                        ; nach A6
        jsr     -90(a6)                 ; DisplayAlert,
                                        ; Fehlermeldung aus-
                                        ; geben
        cmp.l   #1,d0                   ; Linke Maustaste
                                        ; gedrückt?
        beq     systemreset             ; Wenn ja, verzweige
        move.l  libbase(pc),a1          ; Base nach A1
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -414(a6)                ; Library schließen
        movem.l (a7)+,d0-d7/a0-a6       ; Register zurück-
                                        ; holen
        and.l   #$80000000,d7           ; Modus ermitteln
        tst.l   d7                      ; Kann das Programm
                                        ; weiterlaufen?
        beq.s   gurufertig              ; Wenn ja, verzweige
systemreset:                            ; weiter
        jmp     $f80000                 ; Einsprung in die
                                        ; Reset-Routine
gurufertig:                             ; weiter
        rts                             ; Fertig
reset:                                  ; Speichermangel
        move.l  buffer(pc),a1           ; Zeiger auf Not-
                                        ; Speicher nach A1
        move.l  #25000,d0               ; Länge nach D0
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -210(a6)                ; Speicher freigeben
        lea     m1(pc),a1               ; Feld nach A1
        lea     m1c(pc),a0              ; Textzeiger nach A0
        bsr     putdata                 ; Daten einfügen
        lea     lib(pc),a1              ; Zeiger auf Intui-
                                        ; tion-Name nach A1
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -408(a6)                ; Library öffnen
        lea     libbase(pc),a0          ; Base sichern
        move.l  d0,(a0)
        lea     text(pc),a0             ; Zeiger auf Fehler-
                                        ; Text A0
        move.l  #$a0,d1                 ; Höhe in Bild-
                                        ; punkten
        move.l  #$30000000,d0           ; Flag setzen
        move.l  libbase(pc),a6          ; Base nach A6
        jsr     -90(a6)                 ; DisplayAlter,
                                        ; Fehlermeldung aus-
                                        ; geben
        move.l  libbase(pc),a1          ; Intuition-Base
                                        ; nach A1
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -414(a6)                ; Library schließen
        movem.l (a7)+,d0-d7/a0-a6       ; Register zurück-
                                        ; holen
        jmp     $f80000                 ; Einsprung in die
                                        ; Reset-Routine
buffer:                                 ; Zeiger auf Not-
        dc.l    0                       ; Speicher
copy:                                   ; Kopier-Routine
        move.l  #25000,d0               ; Speichergröße nach
                                        ; D0
        moveq   #2,d1                   ; Speichertyp =
                                        ; ChipRam
        move.l  4,a6                    ; ExecBase nach A6
        jsr     -198(a6)                ; Speicher anfordern
        move.l  d0,buffer               ; Zeiger merken
        move.l  #copy,d0                ; Anzahl der zu
        sub.l   #start,d0               ; kopierenden Bytes
        add.l   #8,d0                   ; ermitteln
        asr.l   #1,d0
        asl.l   #1,d0
        move.l  4,a6                    ; ExecBase nach A6
        clr.l   d1                      ; Speichertyp =
                                        ; beliebig
        jsr     -198(a6)                ; Speicher anfordern
        move.l  d0,speicher             ; Zeiger merken
        move.l  d0,a1                   ; Zeiger nach A1
        lea     start(pc),a0            ; Kopierstart
                                        ; nach A0
copy_loop:                              ; Schleife
        move.b  (a0)+,(a1)+             ; 1 Byte kopieren
        cmp.l   #copy,a0                ; Ende erreicht?
        bne.s   copy_loop               ; Wenn nein, dann
                                        ; Schleife
        rts                             ; Fertig
speicher:                               ; Zeiger auf
        dc.l    0                       ; Speicher für
                                        ; Programmcode
```

Guru-Off



Edgar Meyzis

Tips & Tricks, nicht nur für M2Amiga

Auch in Hochsprachen lassen sich residente Programme schreiben, resident im Sinne von DOS. Das Betriebssystem des Amiga (Exec) bietet Mittel, Code resident so in das System einzubinden, daß sogar ein Reset überstanden wird. Davon soll hier nicht die Rede sein, sondern von den Möglichkeiten des DOS, Programme resident zu laden und wiederholt aufzustarten ohne Zugriff auf Speichermedien, den ursprünglichen Quellen der Programme.

Going resident

Haben Sie schon einmal ein Programm resident geladen? Nein, dann nichts wie ran an den Speck. Mit "resident C:run pure" laden Sie z.B. das CLI-Programm "run" resident. Der Zusatz "pure" ist nur erforderlich, falls das "Pure-Bit" in der "Protection Mask" nicht gesetzt ist. Mit der Eingabe von "resident" können Sie sich nun überzeugen, daß "run" tatsächlich als geladen und unbenutzt verfügbar ist. Hingegen wird "resident" selbst als benutzt ausgewiesen. Es wird ansatzweise deutlich, daß DOS einen "Verwaltungsmechanismus" bietet, Programme einfach zu laden und mehrfach, quasi parallel, ablaufen zu lassen.

Die meisten CLI-Befehle sind resident ladbar. Was hat man davon? Vor Aufruf eines Programms sucht DOS die Verzeichnisse entlang des Pfades ab. Vorher wird jedoch die "Resident-Liste" analysiert. Ist das Programm dort enthalten, so wird es einfach aufgestartet. Der Ladevorgang entfällt. Bei Einsatz einer RAM-Disk kommt man gleichfalls ohne Zugriffe auf externe Speichermedien aus. Der Unterschied zum Verfahren "DOS-Resident" besteht darin, daß das Programm aus der RAM-Disk vor dem Aufstart in den Speicher kopiert wird. Es befindet sich dann überflüssigerweise doppelt im Speicher. (Und wer lebt denn schon im Überfluß?)

Mitunter ist es zweckmäßig, eigene Programme "DOS-Resident" zu schreiben, um z.B. lange Ladezeiten zu unterbinden. Mit M2Amiga scheint das ein Kinderspiel zu sein. Es muß lediglich dafür gesorgt werden, daß bei jedem Programmaufstart die initialisierenden Teile im Prinzip vollständig durchlaufen werden. Bei den eigenen Modulen können wir leicht dafür sorgen, bei den Bibliotheksmodulen (z.B. InOut, Heap, Conversions) sind wir darauf angewiesen, daß die Compilerbauer unsere Absicht unterstützen. (Es müßte auch die Ihre sein.) Was heißt das nun konkret, warum die Einschränkung "im Prinzip"?

Modula-2 bietet die Möglichkeit, Implementationsmodule mit einem Rumpf zu versehen, dessen Code nur einmal ausgeführt wird, nämlich unmittelbar nach dem Laden des jeweiligen Moduls. Diese initialisierenden Teile eines Moduls dürfen keinen Code ausführen, der bei jedem Programmaufstart durchlaufen werden muß, um zum Beispiel Variable in Abhängigkeit von bestimmten Zuständen zu initialisieren, Ressourcen zu allozieren (z.B. Speicheranforderung) oder Dateien zu öffnen. Für bestimmte Variablen ist es jedoch durchaus o.k., wenn sie nur einmal initialisiert werden. Darunter fallen Variable, die nach der Initialisierung nur noch lesend benutzt werden. Beispiele dafür sind Tabellen in Form von Feldern (ARRAY), die Namen enthalten, auf die über einen Index nur lesend zugegriffen werden soll, um Namensvergleiche durchzuführen.

Fazit: Global angelegte Variablen, deren Inhalte während der Programmlaufzeit schreibend verändert werden, sind bei jedem Aufstart zu initialisieren. Die Implementationsmodule dürfen in ihren Rümpfen keinen Code enthalten, der zum eigentlichen Programm gehört.

Werfen wir noch einen Blick in die "DOS-Resident"-Welt und stellen uns die Aufgabe, aus einem Programm heraus festzustellen, ob ein bestimmtes Modul resident geladen ist. Die Funktionsprozedur "FindDosResident" (Listing 1) erfüllt die Aufgabe. Dazu ein paar Erläuterungen: Die Struktur, mit der DOS resident geladene Programme verwaltet, müssen wir selbst deklarieren (DosResList). Der Aufbau der Struktur ist bisher von Commodore nicht veröffentlicht. Es könnte sein, daß das Feld "rlSegName" wesentlich länger dimensioniert ist, als hier deklariert. Den Zugriff auf die erste "DosResList" erhalten wir über die DOS-Library", die uns einen Zeiger auf den "RootNode" liefert. Wir hangeln uns in einem Statement mit Hilfe von Zeigern bis zu "DosInfo.netHand" durch. Entgegen vieler Veröffentlichungen weist "DosInfo.netHand" auf das erste von DOS resident gehaltene Programm (genauer auf die dafür erforderliche Verwaltungsstruktur des Typs DosResList).

"DosInfo.netHand" ist auch nicht vom Typ ADDRESS, wie häufig behauptet, sondern vom Typ BPTR (Langwortadresse, geteilt durch vier).

Das Feld "rlSegName" enthält den Namen eines resident geladenen Moduls, jedoch nicht in der allgemein gewohnten Form. Der Namensvergleich beginnt daher am zweiten Byte und erstreckt sich über die Anzahl der Bytes, die als Länge des Namens dem ersten Byte entnommen wird. Wurde das gesuchte Modul als resident eingetragen gefunden, dann liefert die Funktionsprozedur einen Zeiger (Typ BPTR) auf die entsprechende Verwaltungsstruktur, sonst null (NIL).

POINTER/ ADDRESS versus BPOINTER/BPTR

Zu den maschinennahen Elementen von Modula gehören "POINTER" und "ADDRESS". Hinter "POINTER" verbirgt sich nichts anderes als ein Zeiger, der auf eine beliebige Struktur gerichtet sein kann, im Extremfall auf ein einzelnes Byte. Dann haben wir es mit einem Zeiger vom Typ "ADDRESS" zu tun. Wie soll man sich das praktisch vorstellen, ohne Kästchen und Pfeile? Ganz einfach, ein Zeiger ist eine Variable, das heißt ein Speicherbereich (an einer Wortgrenze), der sich über vier Bytes erstreckt. Die Zeigervariable enthält als Wert die Anfangsadresse des referenzierten Objektes, auf das der Zeiger gerichtet ist. Was macht der Compiler daraus, wenn auf ein Objekt über einen Zeiger zugegriffen werden soll? Er lädt den Inhalt des Zeigers in ein Adreßregister und greift mit indirekter Adressierung auf das Objekt zu.

Schauen wir doch einfach dem Compiler zu. Listing 2 enthält ein einfaches Beispiel. Unsere Zeigervariable trägt den Namen "zeiger". Sie wird in der ersten Zeile des Programms auf "feld" gerichtet. Die beiden sich anschließenden Assemblerbefehle zeigen den generierten Code. Nun soll die erste Integerzahl des Feldes um 25 erhöht werden. Dazu wird der Inhalt von "zeiger" nach a3 geladen und die Addition bei indirekter Adressierung durchgeführt usw.

Sollte Sie die Analyse von Code interessieren, dann beschaffen Sie sich doch die Public-Domain-Disk "Amok 12", die das Programm "DisObject" enthält. Noch komfortabler geht es mit "DecObj", einem Werkzeug aus der Serie M2Amiga, das neben dem Assemblercode auch noch den ursprünglichen Quelltext in Modula-2 anzeigen kann.

Nun ist der "BPOINTER" dran. Das DOS unseres Amiga hat uns BCPL-Zeiger beschert, die nicht die wirkliche Adresse eines Objektes enthalten, sondern die durch vier geteilte. BCPL-Zeiger sind somit letztlich immer auf Langwortadressen gerichtet. Das DOS benötigt diese Form der Adreßangabe als Zugeständnis an die Programmierung in BCPL. Anwendungsbeispiele sind Zeiger auf "FileHandle" und "FileLock".

Der Compiler des M2Amiga bietet die Möglichkeit, BCPL-Zeiger als "BPOINTER" zu realisieren. Der Zeiger auf ein "FileHandle" ist nichts ande-

res als ein "BPOINTER TO FileHandle". Über einen "BPOINTER" kann von Modula aus problemlos auf das referenzierte Objekt zugegriffen werden. Die Adreßumwandlung nimmt der Compiler ab. Sie ist nicht durch ein doppeltes SHIFT oder ähnliche Tricks explizit zu programmieren. Sollte einmal eine Typkonversion von "BPOINTER" nach "ADDRESS" oder von "POINTER" nach "BPTR" notwendig sein, dann paßt der Compiler die Adressen automatisch an. Listing 3 liefert entsprechende Beispiele. Ein "CAST" würde nicht zu dem gewünschten Ergebnis führen, weil eine Adreßanpassung unterbleibt. Ein Zeiger vom Typ "BPTR" entspricht dem Typ "ADDRESS" und verfügt über dieselbe Universalität. Das bedeutet, daß der Typ "BPTR" mit allen Zeigervariablen kompatibel ist, die mit "BPOINTER" deklariert wurden. Darüber hinaus ist noch eine Kompatibilität zum Typ "LONGINT" gegeben. Listing 3 veranschaulicht die Ausführungen und deckt auf, daß bei Zuweisung der Adresse eines Objektes an eine Variable des Typs "BPOINTER" die Adreßkonversion selbständig erfolgt. Ist das nicht toll? (mm)

Listings

```
PROCEDURE FindDosResident (name : ARRAY OF CHAR) : BPTR;
TYPE
     ResListPtr = BPOINTER TO DosResList;
     DosResList = RECORD
                     rlNext     : ResListPtr;
                     rlUsecount : LONGINT;
                     rlSegList  : BPTR;
                     rlSegName  : ARRAY[0..25] OF CHAR;
                  END;

VAR resPtr  : ResListPtr;
    dosLib  : DosLibraryPtr;
    laenge  : INTEGER;
    charPtr : POINTER TO ARRAY[0..25] OF CHAR;

BEGIN
   dosLib := ADR (Dos);
   resPtr := CAST(ResListPtr, dosLib^.root^.info^.netHan
d);
   laenge := Length (name);
   WHILE resPtr # NIL DO
      WITH resPtr^ DO
         IF ORD (rlSegName[0]) = laenge THEN
            charPtr := ADR (rlSegName);
            IF Compare
               (charPtr^, 1, laenge, name, FALSE) = 0 TH
EN
               RETURN resPtr
            END;
         END;
         resPtr := rlNext;
      END;
   END;
   RETURN resPtr;
END FindDosResident;

Listing 1: Einblick in die Verwaltung resident geladener
           Programme
```

```
MODULE PointerShow;

FROM SYSTEM IMPORT ADR;

VAR zeiger : POINTER TO INTEGER;
    feld   : ARRAY[0..5] OF INTEGER;

BEGIN
  zeiger := ADR (feld);

    LEA     -18(A4), A3
    MOVE.L  A3, -6(A4)

  zeiger^ := 25;

    MOVE.L  -6(A4), A3
    MOVE.W  #25, (A3)

  INC (zeiger, 4);

    ADDQ.L  #4, -6(A4)

  zeiger^ := 16;

    MOVE.L  -6(A4), A3
    MOVE.W  #16, (A3)

END PointerShow.

Listing 2: So einfach arbeiten POINTER ?!
```

```
MODULE BPointerShow;

FROM SYSTEM IMPORT BPTR, ADDRESS, ADR;

VAR b1  : BPOINTER TO INTEGER;
    p1  : POINTER TO INTEGER;
    int : INTEGER;

BEGIN
  b1  := ADR (int);

    LEA     -12(A4), A3
    MOVE.L  A3, D6
    LSR.L   #2, D6
    MOVE.L  D6, -6(A4)

  b1^ := 20;

    MOVE.L  -6(A4), A3
    ADDA.L  A3, A3
    ADDA.L  A3, A3
    MOVE.W  #20, (A3)

  p1  := ADDRESS (b1);

    MOVE.L  -6(A4), D6
    ASL.L   #2, D6
    MOVE.L  D6, -10(A4)

  b1  := BPTR (p1);

    MOVE.L  -10(A4), D6
    LSR.L   #2, D6
    MOVE.L  D6, -6(A4)

END BPointerShow.


Listing 3: BCPL-Zeiger effizient
           implementiert
```

fischer
Hard- und Softwareversand
Leddinweg 14
3000 Hannover 61
Tel. 0511-572358
Fax/Btx 0511/572373
Wir sind Colossus Distributor – Händleranfragen erwünscht.
Wir bieten Ihnen mehr als nur ein gutes Angebot. Unser Service: Bestellungen und Beratungen bis 20 Uhr. Schnelle Lieferung. Lagerware verläßt am nächsten Tag unser Haus.

Tun Sie Ihrem Computer was gutes mit unserem Zubehör
A500 512 KB Uhr, Abschalter 189,-
A2000 2 MB auf 8 MB Platine 995,-
Laufwerk 3,5" extern 198,-
Laufwerk 3,5" intern 148,-

neu NEC P2 plus neu
998,-
schneller und leiser als der P2200, billiger als der P6 plus.
Der richtige Drucker für Sie. Ja Sie!!!

ColoSsus Filecards
bis 500 KB/sec
32 MB 28 ms 1198,-
20 MB 28 ms 998,- 43 MB 19 ms 1498,-
47 MB 28 ms 1498,- 66 MB 19 ms 1798,-
sofort betriebsbereit, autoboot ab Kickstart 1.2, lauffähig mit und ohne PC/AT/68020-Karte. Colossus-Festplatten-Backup-Programm kostenlos im Lieferumfang enthalten.
Sie machen keine Sicherheitskopie von Ihrer Festplatte, das kann Folgen haben, was Sie brauchen ist DAS schnelle
Colossus-Harddisk-Backup-Programm für 79,–

Commodore 68020-Karte incl.
68881/68851 und 2 MBRam, Unix-fähig
2745,-

AT-Karte 1998,- PC-Karte 775,-
Nec P6 plus 1398,-
Farb-Option P6 plus 288,-
2090 A Controller 598,-
40 MB + 2090 A 1498,-
Nec Multisync 3 D 1598,-
Amiga 2000 Komplettsysteme zu Superpreisen auf Anfrage z.B. A2000 + 47 MB Colossus 3300,-

Frank Schäfer

Logische Devices Alles Logo?

Aufgaben, Anwendungen, Tips und Tricks

AmigaDOS-Einsteiger werden sicherlich über Bezeichnungen, die mit einem Doppelpunkt versehen sind, gestolpert sein. Sofern es sich um Diskettenlaufwerke oder die RAM-Disk handelt, ist ja alles im klaren, doch schon beim Verzeichnis c fängt das Grübeln an: Warum steht da bloß ein Doppelpunkt dahinter? Was es mit diesen sogenannten Devices bzw. virtuellen Geräten auf sich hat, können Sie im folgenden Artikel erfahren.

Übersetzt bedeutet Device soviel wie Gerät oder Vorrichtung. Unter einem logischen Device versteht man also ein logisches Gerät, gewöhnlich ein Programm. Im Amiga-Betriebssystem ist ein Device eine Sammlung von Routinen und Daten, die zur Kommunikation des Systems mit der Außenwelt benötigt werden.

Warum eigentlich Devices

Auf der Workbench-Diskette existieren Directories mit Namen wie c, libs, devs, fonts, etc. und weiterhin gibt es im System auch die Devices c:, libs:, devs:, fonts: usw.

Da stellt sich doch sofort die Frage, warum Devices den gleichen Namen besitzen, wie Directories auf der Workbench. Zur Beantwortung dieser Frage stellen wir uns einmal vor, es gäbe diese Devices nicht. Falls wir nun von einem Programm aus einen Zeichensatz laden wollen, könnte dieses Programm beispielsweise auf "df0:fonts" zugreifen. Nun möchten wir aber Zeichensätze benutzen, die sich auf der Harddisk oder in einem anderen Laufwerk befinden. Dies wäre nach der eben beschriebenen Methode nicht, oder nur umständlich möglich. Zur Lösung dieses Problems wurden die Devices geschaffen. Sie bieten die Möglichkeit, dem System auf einfache Art und Weise mitzuteilen, von wo ein Zeichensatz oder Befehl, ein Handler oder ein Programm geladen werden soll.

Mittels des Assign-Befehls kann man die Zuweisung von logischen Devices (Laufwerk, Gerät, Programm) verändern. Weiterhin kann man mit diesem Befehl neue logische Devices erzeugen oder löschen. Auch die Anwendung als ALIAS-Befehl ist möglich.

Syntax des Assign-Befehls

1. ASSIGN [List]

Falls man den Befehl "Assign" ohne Zusatz eingibt, erhält man eine Liste, in der sämtliche Diskettennamen, Devices und Laufwerke enthalten sind. Diese Liste ist in "Volumes:", "Directories:" und "Devices:" unterteilt.

In dem Volumes-Teil sind die Namen aller dem System bekannten Disketten aufgeführt. Die Zusatzbezeichnung "MOUNTED" bedeutet dabei, daß sich die bezeichnete Diskette in einem der Laufwerke befindet. Der Name einer Diskette (und auch das Disketten-Icon auf der Workbench) verschwindet übrigens erst dann aus dieser Liste, wenn kein Lock mehr auf sie weist. Unter einem Lock ist ein Pointer zu verstehen, der auf ein bestimmtes File/Directory der Diskette zeigt. So ist zum Beispiel auch ein offenes Window auf der Workbench, welches zu der betreffenden Diskette gehört, ein Lock.

Unter "Directories:" sind alle logischen Devices aufgeführt, die zu einem bestimmten Directory/File gehören. Diese Bezeichner kann man auch löschen (siehe Position 3).

Bei "Devices:" werden die Laufwerke angegeben, die einen eigenen Proze bzw. Handler ihr eigen nennen können. Hierzu zählen natürlich alle Diskettenlaufwerke (deren Prozeß nennt sich immer "File System"), alle Treiber für Schnittstellen (PAR, PRT, SER, CON, RAW, ...) sowie alle sonstigen Laufwerke wie beispielsweise Hard- und RAM-Disks.

2. ASSIGN DeviceName: [DeviceName:] PfadName {/PfadName}

Diese Syntax des Assign-Befehles bedeutet folgendes: Das virtuelle Laufwerk "DeviceName:" wird in Zukunft im AmigaDOS unter dem Pfad "[DeviceName:] PfadName {/PfadName}" erreicht. Die etwas kryptische Schreibweise findet man im AmigaDOS sehr häufig. Falls man "Befehl?" eingibt, erhält man praktisch immer eine Erklärung zu dem betreffenden Befehl, die allerdings häufig in EBNF (EBNF = Erweiterter Backus Naur-Formalismus) abgefaßt ist. Box 1 erläutert den EBNF ausführlicher.

In der oben gezeigten Syntax kann der Assign-Befehl zwei Dinge bewirken: Zum einen kann man den DOS-Pfad, unter dem das virtuelle Device zu erreichen ist, verändern, und zum anderen können neue Devices erzeugt werden.

3. ASSIGN DeviceName: [remove]

Die Anwendung des Assign-Befehls in der oben gezeigten Form auf ein logisches Device (Directory) bewirkt, daß dieses gelöscht wird. Das Löschen von Devices, die eine Hardwareentsprechung haben, ist aus verständlichen Gründen nicht möglich. Ein "Assign c:" löscht als das Device c:. (Anm. der Red.: Sehr unangenehm)

4. ASSIGN DeviceName: [DeviceName:] {/PfadName} FileName

Die Befehlsnamen im CLI sind oft recht lang. Mit Hilfe des Assign-Befehls in der gezeigten Form kann man sich jedoch viel Tipparbeit ersparen. Um den Execute-Befehl durch "x:" zu ersetzen, benötigt man lediglich folgende Eingabe:

assign x: c:execute

Ein Batchfile kann nun durch "x: Name" gestartet werden. Diese Form der Abkürzung von Befehlen ist eleganter und benutzerfreundlicher als die Namensveränderung durch den Rename-Befehl.

Einige Shells (Shell, zu deutsch Muschel, bedeutet in der Computersprache so viel wie ein komfortables CLI) bieten dem Benutzer einen Alias-Befehl an. Durch die Anwendung des Assign-Befehls in dieser Form kann man den Alias-Befehl ersetzen und auch das CLI etwas aufwerten.

Was passiert beim Booten mit den Geräten

Auf einer normalen Workbench-Diskette befinden sich, wie bereits erwähnt, Directories mit den Namen c, devs, l, libs, s und fonts. Beim Booten vom Laufwerk df0: führt das System sinngemäß für jedes angegebene Directory dann folgende Befehle aus:

"assign c: df0:c",
"assign devs: df0:devs",
...

Falls sich beispielsweise das Devs-Directory nicht auf der Bootdiskette befindet, wird der Befehl modifiziert:

"assign devs: df0:".

Folglich sucht das Betriebssystem das File System-Configuration, das beim Booten benötigt wird, direkt auf dem Startlaufwerk df0:. Entsprechendes gilt natürlich auch für die anderen Directories.

Zuerst einmal eine Übersicht über die logischen Devices, die das System beim Booten anlegt.

c:

Hier werden standardmäßig die CLI-Befehle gesucht. Der Grund, warum die CLI-Befehle dort gesucht und gefunden werden, ist im PATH-Befehl begründet. Bei der Initialisierung eines CLI-Prozesses werden verschiedene Voreinstellungen durchgeführt. Zu diesen Voreinstellungen gehört auch das Eintragen der Pfadliste in die Prozeßdatenstruktur des CLI. Diese Pfadliste beinhaltet zwei Eintragungen: das aktuelle Directory und c:. Da das Laden von der Diskette einige Zeit beansprucht, bietet es sich an, diesen Vorgang etwas zu beschleunigen, indem man die CLI-Befehle ins RAM kopiert. Eine übliche Vorgehensweise bei der Erstellung einer RAM-Disk oder beim Einrichten einer Harddisk sieht wie folgt aus:

makedir ram:c

Es wird ein Directory erstellt, in dem die CLI-Befehle abgelegt werden.

copy c: ram:c

Die CLI-Befehle werden nun in das neue RAM-Verzeichnis c: kopiert.

assign c: ram:c

Ab jetzt sucht das CLI seine Befehle im RAM, denn der Pfad, unter dem das Device zu erreichen ist, wurde geändert.

Außer dieser Vorgehensweise gibt es natürlich noch andere Methoden, wie zum Beispiel der Resident-Befehl, AddBuffers oder FaccII.

An dieser Stelle soll einmal etwas zur Begriffsklärung getan werden. Das Beriebssystem des Amiga kennt zwei Datenstrukturen, die für parallel laufende Prozesse benutzt werden. Zum einen die Task-Struktur und zum anderen die Prozeß-Struktur. Eine Prozeß-Struktur ist eine um einiges erweiterte Task-Datenstruktur. Ein Task ist gegenüber einem Prozeß in einer wesentlichen Funktion eingeschränkt, er kann keine Funktionen der DOS-Library benutzen. Da im CLI aber immer (!) DOS-Funktionen benötigt werden, ist ein CLI als Prozeß und nicht als Task zu verstehen.

devs:

In diesem Device befinden sich üblicherweise drei weitere Directories, mit den Namen PRINTERS, CLIPBOARDS und KEYMAPS. Im Directory PRINTERS wird der im File SYSTEM-CONFIGURATION angegebene Drucker-Treiber gesucht. Das File SYSTEM-CONFIGURATION wird von dem Programm Preferences erzeugt. In ihm sind alle Voreinstellungen (Mauszeiger, Drucker, Farben, Schnittstellen...) abgelegt, die man in Preferences vorgenommen hat. Das Directory CLIPBOARDS wird von ClipBoard-Device benötigt, um dort Clips (beliebige Daten, Grafik) zwischenzuspeichern. Das Clipboard-Device ist genau wie das Printer-Device oder das Parallel-Device als ein Device im Sinne eines logischen Gerätes (Programm) zu verstehen. Das bedeutet, daß man zu einem solchem Device Daten senden und/oder von ihm empfangen kann. Das Directory KEYMAPS wird vom SETMAP-Befehl benötigt. Dieser sucht dort die als Parameter angegebene Keymap (Tastatur-Belegung). Im logischen Device "devs:" befindet sich auch das immer wichtiger werdende File Mountlist. Dieses wird benötigt, um dem System mittels des MOUNT-Befehls mitzuteilen, welche zusätzlichen Geräte auf Softwarebasis (Speaker, NewCon, Aux) bzw. Hardwarebasis (Harddisks, Laufwerke, Wechselplatten, CD-ROMs) vorhanden sind. Auch das zur Sprachausgabe benötigte Narrator-Device befindet sich in Devs:. Kurz gesagt, im Device devs: befinden sich Programme und Daten, die zur Verwaltung von externen Geräten benötigt werden.

l:

In diesem Device findet man Handler (Programme), die zum Betrieb von internen Geräten benötigt werden. Diese Geräte sind auch wieder als Programme zu verstehen. In der Mountlist wird angegeben, welcher Handler zu welchem Gerät gehört. Eine Ausnahme macht hier unter anderem der Disk-Validator. Er wird vom DOS automatisch geladen, wenn eine Diskette als nicht validiert erkannt wird. Prinzipiell wird eine Diskette als solche nicht erkannt, wenn die BitMap (ein spezieller Datenblock) der Diskette mit dem aktuellen Füllzustand der Diskette nicht übereinstimmt. Auf einer bootfähigen Diskette sollte sich immer ein Disk-Validator befinden. Auch die Handler von SHELLs und resetfesten RAM-Disks gehören in dieses Device.

t:

Dieses Device wird von vielen Programmen zum Zwischenspeichern von Daten benutzt. Man sollte in der Startup-Sequence (siehe s:) immer ein ASSIGN t: ram:t vornehmen (vorher aber im RAM mit "makedir ram:t" ein Directory "t" erzeugen). Programme, die ein Device mit dem Namen t: benutzen, sind z.B. TxED (legt ein Backup des Sourcecodes dort ab) oder Word Perfect (Zwischenspeichern von Daten und Blöcken) Batchdateien etc.

fonts:

Hier suchen alle Programme nach den Zeichensätzen, die verwendet werden sollen. Durch ein Umlegen dieses Devices auf ein anderes Laufwerk/Directory ist es recht einfach möglich, neue Zeichensätze zu benutzen. Da das Ändern des font-Devices eine der häufigsten Anwendungen des Assign-Befehles ist, soll dies anhand des Grafikprogramms Deluxe-Paint II genauer beschrieben werden. Wir wollen den Fall annehmen das Deluxe Paint II bereits gestartet wurde. Zum Auswählen eines Zeichensatzes ist im Menü "FONT" die Anwahl der Menüposition "Load Font Dir" erforderlich. Nach einiger Zeit hat Deluxe Paint II dann den Inhalt des Font-Devices gelesen und stellt ihn als Untermenü im "Font-Menü" dar. Nun ist das Auswählen eines Zeichensatzes möglich.

ASSIGN – Der Befehl, der keine Fragen offen läßt

Mittels der Taste "T" oder des A-Gadgets gelangt man in den Texteingabemodus, der durch Drücken von "ESC" oder einer anderen Gadget-Funktion wieder verlassen werden kann. Falls man andere Zeichensätze benutzen möchte, muß man zuerst einmal ins CLI. Deluxe Paint schließt, falls kein CLI geöffnet ist, die Workbench, mittels der CloseWorkBench-Funktion. Falls dies geschehen ist, kann man mit der Funktion "Workbench" im Menü "Prefs" die Workbench öffnen. Im CLI angelangt, gibt man nun sinngemäß folgendes ein:

ASSIGN fonts: [DFx:]{Pfad/} DirectoryName

Den Namen des Directories, in dem sich die Zeichensätze befinden, kann durch den DIR-Befehl in Erfahrung gebracht werden. Gewöhnlich handelt es sich aber um ein Directory mit dem Namen Fonts. Der Assign-Befehl sieht dann wie folgt aus:

ASSIGN fonts: DF1:Fonts

Wieder in der Deluxe Paint-Umgebung, wählt man die Option "Screen Format" im "Project-Menü" an. In der nun auftauchenden Popup-Form (Window/Requester mit Gadgets) klickt man "Ok" an. Durch diese Aktion wurde der Inhalt des Fontmenüs gelöscht und kann nun mit "Load Font Dir" neu geladen werden. Da das logische Device fonts: inzwischen auf ein anderes Laufwerk/Directory umgelegt wurde, wird dessen Inhalt angezeigt.

Der interne Aufbau eines Font-Directories sieht nun so aus: Für jeden Zeichensatz existiert ein File mit dem Namen xxx.font; dieses File enthält Grunddaten zum jeweiligen Zeichensatz. In dem zu jedem Font gehörenden Directory mit dem Namen xxx stehen Files, deren Namen der Höhe der Zeichensätze in Punkten entsprechen. In ihnen sind für den jeweiligen Zeichensatz die Grafikdaten der Zeichen abgelegt.

s:

Dieses Device wird hauptsächlich für Batch-Dateien benötigt. Die wichtigste Datei in diesem Device ist die Startup-Sequence. Dieses Batch-File wird vom Betriebssystem beim Booten automatisch abgearbeitet. In der Startup-Sequence werden Harddisks und/oder resetfeste Ramdisks installiert, die Workbench geladen, das CLI geschlossen, Zeichensätze ausgewählt, der Stack verändert und Tools gestartet. Jeder Amiga-User sollte seine Startup-Sequence kennen und verändern können. Eigene Batch-Dateien, die zum Beispiel einen Compiler oder eine Textverarbeitung installieren und starten, sollte man auch in diesem Device abspeichern.

libs:

Dieser Device-Name ist eine Abkürzung des Wortes Library. Eine Library, zu deutsch Programm-Bibliothek (Sammlung), besteht aus vielen Unterprogrammen, die über eine Sprungtabelle angesprungen werden. In der Kickstartversion 1.2 sind acht verschiedene Libraries integriert. Da im Kickstart-ROM (WOM bei A1000) nicht genügend Speicherplatz vorhanden war, wurden einige Libraries auf Diskette ausgelagert. Diese Libraries seien an dieser Stelle aufgeführt.

- diskfont.library
  (enthält Datenstrukturen und Funktionen, um Zeichensätze von der Diskette zu laden)
- icon.library
  (enthält Prozeduren zum Arbeiten mit Icons)
- info.library
  (hier sind Diskinfos zu finden)
- mathieeedoubtrans.library
  (enthält Winkel- und logarithmische Funktionen für LONGREAL-Zahlen (MC68881 unter 1.3))
- translator.library
  (ist für die Sprachumwandlung zuständig)

- mathieeedoubbas.library (unterstützt die Grundrechenarten für LONGREAL-Zahlen (MC68881 unter 1.3))
- mathtrans.library (enthält Winkel- und logarithmische Funktionen für FFP-Zahlen)
- version.library (Versionsabfrage für Kickstart, Workbench, und Libraries)
- xxx_lib.fd (Das AmigaBASIC erwartet, daß in dem Device libs: seine FD-Files zu finden sind)

Hierzu kommen noch einige Libraries von Anwendungsprogrammen, die gewöhnlich aber nicht dokumentiert sind. Eine Ausnahme bilden hier die Libraries von ARP, ARexx und FaccII. Die Libraries von Word Perfect oder Project D und anderen Programmen muß man, falls man das Programm aufruft, im aktuellen libs-Device haben. Sprich, wenn das Programm von der Workbench gestartet wird, müssen sich die entsprechenden Files auch im libs-Directory der Workbench befinden. Da man aber auf der Workbench gewöhnlich kaum Platz hat, kann man mittels IconX (früher XIcon) das libs-Device auf die WP-Disk umlegen. (IconX ist ein Programm, mit dem man von der Workbench aus Batchdateien starten und ausführen lassen kann.)

Auch hier wieder einige Sätze zur Begriffsklärung: Durch das Betriebssystem des Amiga werden dem Programmierer zirka 450 Funktionen bereitgestellt. Die Funktionen sind nach ihren Anwendungs- und Aufgabengebieten in verschiedene Libraries aufgeteilt. Diese Libraries sind teilweise im WOM/ROM und auf Diskette zu finden. Falls eine benötigte Library sich auf der Diskette befindet, wird diese vom Betriebssystem automatisch geladen.

Libraries bestehen aus einem festen und einem veränderlichen Teil. Bei der Systeminitialisierung wird für den veränderlichen Teil Speicher reserviert. Der veränderliche Teil der Library besteht grundsätzlich aus zwei Teilen. Zum einen dem Datenbereich der Library, der im Speicher oberhalb der Basisadresse liegt, und zum anderen der Sprungtabelle auf die einzelnen Funktionen, die im Speicher unterhalb der Basisadresse steht. Da in der Library-Datenstruktur ein Zähler vorhanden ist, der bei jedem Öffnen der Library um eins inkrementiert und bei jedem Schließen um eins dekrementiert wird, sollte man Libraries immer ordentlich öffnen und schließen. Wenn der OpenCounter der Library durch das Schließen auf Null gesetzt ist, kann die Library aus dem System entfernt werden, um somit Speicherplatz zurückzugewinnen.

sys:

Dieses Device entspricht der Bootdiskette. Ein DIR-Befehl auf sys: entspricht folglich einem DIR auf die Diskette von der gebootet wurde; unter der KickStart-Version 1.2 ist das praktisch immer DF0:. Mit der KickStart 1.3 kann man aus dem RAM booten. Die logische Folge davon ist sys: zeigt dann auf card:. Die Nützlichkeit von sys: zeigt sich beim Erstellen von Startup-Sequencen und Batch-Files. Man kann unabhängig vom Diskettennamen und Laufwerk die Bootdiskette direkt ansprechen. Falls man eine Harddisk besitzt, sollte man in der Startup-Sequence sys: auf die Harddisk "umassignen". (Assign sys: dh0:). Beim Kopieren erweist sich sys: als äußerst nützlich. Ein kleines Beispiel soll dies verdeutlichen:

copy xxx sys:system

Falls die Bootdiskette sich in keinem Laufwerk befindet, wird sie vom DOS angefordert. Angenommen, sie befindet sich in einem beliebigen Laufwerk, wird dieses automatisch durch das System erkannt und der Copy-Befehl ausgeführt. Dies ist allgemein gültig. Dir, List, Run, Delete, usw. arbeiten entsprechend.

ram:, dfx:, dhx:, vd0:, card:,...

Die oben aufgeführten Devices werden, im Gegensatz zu den bis jetzt besprochenen, häufig als "echte" Laufwerke betrachtet und auch so benutzt. Dabei erkennt der Benutzer jedoch aufgrund der internen DOS-Struktur keinen Unterschied zwischen der Befehlszeile

copy c:dir ram:

und der Zeile

copy df0:c/dir ram:

falls c: noch auf die Diskette in df0: "assigned" ist.

Nach der EBNF-Notation ist eine Variable (egal ob in Basic, C, Modula, Pascal oder Assembler) definiert als:

ident = letter{letter|digit}

Übersetzt heißt das: Eine Variable besteht aus mindestens einem Buchstaben. Das erste Zeichen einer Variablen ist folglich immer ein Buchstabe. Sie kann sich jedoch beliebig in Form einer Buchstaben-Zahlenkombination: z.B. A, AB5, Wert24, H2OKopf oder TAL fortsetzen. Durch den Zusatz ist die geschweifte Klammer als eine beliebig lange Aneinanderkettung ihres Inhalts definiert. Das Metazeichen (oder auch Pipe-Symbol) "|" bedeutet dabei nichts anderes als 'dieses oder jenes', sprich das exclusive ODER.

Die beiden anderen wichtigen Symbole der EBNF-Notation sind die eckigen und die runden Klammern. Mit den folgenden Beispielen soll ihre Anwendung kurz erläutert werden.

octalDigit = |"0"|"1"|"2"| "3"|"4"|"5"|"6"|"7"

Hiermit sind die Oktalzahlen (Zahlensystem mit nur acht Ziffern) definiert. Eine Ziffer dieses Systems kann entweder eine Null oder eine Zahl zwischen eins und sieben sein.

digit = octalDigit|"8"|"9"

Diese Definition steht für das (normale) Zehnerzahlensystem. Das Sechszehner-Zahlensystem (Hexdezimalsystem) wird durch

hexDigit = digit|"A"|"B"| "C"| "D"|"E"|"F"

zugewiesen.

integer = digit{digit}| octalDigit{oktalDigit} ("B"|"C")| digit{hexDigit}"H"

Diese Definition ist in Modula oder Pascal zu finden, wobei die obige Zeile die Integerzahl definiert. Da eine Integerzahl in jedem Zahlensystem dargestellt werden kann, erfolgt auch die Definition der Zahl in den drei wichtigen Systemen. In anderen Sprachen werden oft andere Zeichen als "B","C" oder "H" verwendet, um die verschiedenen Zahlensysteme zu definieren. Im Oktal-System könnte eine Zahl dann wie folgt aussehen:

"34565B" bzw. "34565C"

und im Hexadezimal-System so:

"0A5CF3H".

Aus der Definition geht hervor, daß am Anfang immer ein Digit stehen muß (eine Ziffer aus dem Zehnersystem, notfalls die Null). Wenn dies nicht der Fall wäre, könnte man Variablen nicht von Hexadezimal-Zahlen unterscheiden. Die wesentliche Erkenntnis aus der Definition der Integerzahl ist aber die Bedeutung der runden Klammer. Mit ihr kann man an ein 'Entweder-Oder' ausdrücken, das aber keine Wiederholungen gestattet, wie es die geschweifte Klammer gestattet. Das heißt, daß man entweder das "B" oder das "C" benutzen muß, wenn man eine Oktalzahl ausdrücken will.

Als letztes zur EBNF-Notation sei hier noch die Erläuterung der eckigen Klammer erwähnt.

Jeder wird sicherlich die 'wissenschaftliche' Schreibweise für Zahlen kennen. "0.0025" kann nach dieser Schreibweise auch als "2.5 * 10^-3" (in Kurzform auch "2.5E-3") geschrieben werden. Wobei man die Potenz auch als SkaleFactor bezeichnet, dieser Faktor ist wie folgt definiert.

SkaleFactor = "E"[" + "|"-"] digit{digit}

Die eckige Klammer bedeutet ähnliches wie die runde Klammer. Da man bei dem Faktor aber das Pluszeichen nicht schreiben muß, bedeutet diese Klammer, daß man entweder ein Element aus der Klammer benutzen kann oder auch nicht.

Man kann EBNF allerdings auch etwas formaler definieren. Dies sähe etwa so aus:

(A|B)(C|D) ^IAC AD BC BD
A[B]C ^IABC AC
A{BA} ^IA ABA ABABA ...
{A|B}C C AC BC AAC ABC ..

Hier noch ein Beispiel, wie man eine Programmiersprache mittels EBNF definiert:

Anweisung:	**Eine syntaktische Gliederung.**
Ausdruck:	**Eine Liste alternativer Terme.**
Term:	**Eine Zusammensetzung von Faktoren.**
Faktor:	**Eine einzige syntaktische Einheit oder ein geklammerter Ausdruck**

Syntax = { Anweisung }
Anweisung =
Name " = " Ausdruck "."
Ausdruck =
Term { "|" Term }
Term =
Faktor { Faktor }
Faktor =

Name|
String | "(" Ausdruck ")" |
"[" Ausdruck "]" |
"{" Ausdruck "}"

Name =
Buchstabe{Buchstabe|Ziffer}
String =
"{Buchstabe|Ziffer}"

Box 1: Hinweise zum EBNF

Spiele auf der Harddisk

Da der Amiga häufig als Spielecomputer benutzt wird, stellen sich natürlich die Fragen: Wie bringt man ein Spiel auf Harddisk, und kann man alle Spiele auf Harddisk installieren?

Auf die zweite Frage kann man mit einem glatten NEIN antworten. Viele Spiele besitzen "Fastloader", eigene Diskettenformate und einen Kopierschutz. Einen Teil der Spiele kann man aber auf der Harddisk installieren. Am Beispiel Shanghai soll einmal gezeigt werden, wie dies geschieht.

Der erste Versuch ist, das Icon des Spiels einfach auf die Harddisk zu legen, in der Hoffnung, daß die Workbench das Spiel richtig kopiert. Ein Versuch, das Spiel zu starten, ist allerdings erfolglos, denn Shanghai vermißt eine Diskette mit dem Namen "Shanghai". Dieses Problem ist einfach dadurch zu lösen, indem man im CLI die folgende Zeile eingibt:

assign Shanghai: dh0:

Dies allerdings nur unter der Voraussetzung, daß man das Spiel direkt auf dh0: kopiert hat. Falls es sich beispielsweise auf der Janus-Harddisk im Verzeichnis Spiele befindet, sähe der Assign-Befehl so aus:

assign Shanghai: jh0:Spiele

Ein erneuter Start bringt wieder eine Fehlermeldung, das Programm vermißt nämlich ein Directory "Winter" und dort ein Bild mit dem Namen "Shanghai.title". Hier hilft folgendes: In dem Directory, in dem sich das Programm Shanghai befindet, erzeugt man ein weiteres Directory mit dem Namen "Winter". In dieses Unterverzeichnis müssen dann die Files hineinkopiert werden, die sich im Directory "Winter" der Originaldiskette befinden.

makedir dh0:winter
copy df0:winter dh0:winter

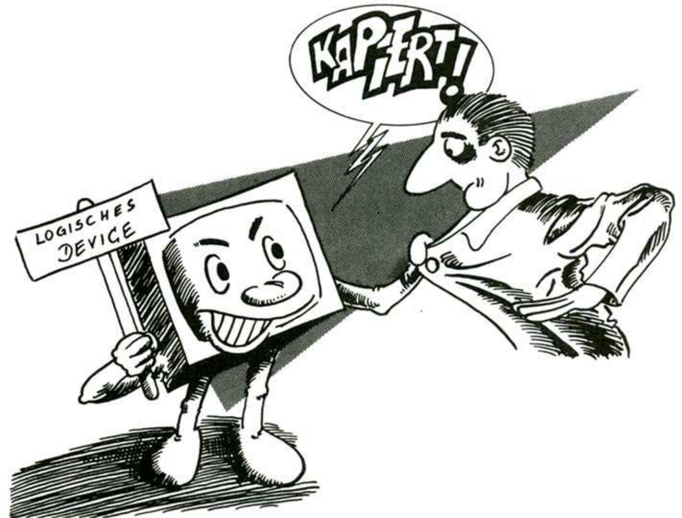
Ein weiterer Start zeigt uns, daß wir noch etwas vergessen haben, ein Zeichensatz (font) mit dem Namen "Gene.font" wird vermißt. Nun diesen Zeichensatz in das aktuelle Fonts-Device der Harddisk hineinkopieren, und das Spiel läuft ... nicht!

Abhilfe schafft hier

makedir fonts:gene
copy df0:fonts/gene fonts:gene
copy df0:fonts/gene.font fonts:

Es wird jedoch noch ein weiterer Font mit dem Namen "garnet" vermißt. Diesen noch auf die Harddisk kopiert (genau wie bei dem Gene-Font), und das Problem ist gelöst.

(br)



Diskettenlaufwerke

3 1/2" Laufwerk AMIGA 2000 intern — DM 159,–
komplett mit Einbaukit und Anleitung
3 1/2" Laufwerk für alle AMIGAs extern — DM 229,–
abschaltbar, Busdurchführung, AMIGAfarben
5 1/4" Laufwerk für alle AMIGAs extern — DM 279,–
abschaltbar, Busdurchführung, 40/80 Tracks, AMIGAfarben

Disketten

3 1/2" No Name 2DD — 10 St. DM 16,95
3 1/2" Verbatim Verex 2DD — 10 St. DM 25,–
3 1/2" Sentinel COLOR-Disk — 10 St. DM 29,90
lieferbar in Grün, Gelb, Rot, Orange und Weiß
5 1/4" No Name 2S2D — 10 St. DM 5,90
5 1/4" Verbatim Verex — 10 St. DM 14,90
Staffelpreise auf Anfrage

Festplatten

Kapazität	Speed	Filecard	A200	A500
20MB/51/4"	65ms	-	798,–	998,–
20MB/31/2"	35ms	898,–	848,–	1048,–
30MB/51/4"	65ms	-	898,–	1098,–
30MB/31/2"	35ms	1148,–	1098,–	1298,–
40MB/51/4"	28ms	-	1198,–	1398,–
50MB/31/2"	35ms	1398,–	1348,–	1548,–
60MB/51/4"	28ms	-	1398,–	1598,–

Alle unsere Festplatten werden wahlweise mit ALF V1.6 oder mit Autoboot-Modul ausgeliefert.

AUTOBOOTMODUL für Amiga

Amiga 2000 Autobootmodul — DM 119,–
Amiga 500 Autobootmodul — DM 149,–
(Bitte Festplatten- und Controllertyp angeben)

Festplatten-Interface — DM 99,–
Die Adapterplatine paßt den PC-BUS eines Festplatten-Controllers an den AMIGA-BUS an. (Bitte Rechnertyp angeben)

ALF 1.6 Festplattentreiber — DM 98,–
ALF 2.0 Festplattentreiber — DM 198,–
MFM-Set — DM 349,–
OMTI 5520B, ALF V1.6, Festplatten-Interface, Kabelsatz
RLL-Set — DM 369,–
OMTI 5527B, ALF V1.6, Festplatten-Interface, Kabelsatz

Software

X-Copy II — DM 49,–
X-Copy II mit Hardwarezusatz — DM 69,–
Sherlock V2.0 AntiVirusTools — DM 49,–
Turboprint II — DM 89,–
RAM-Test für Amiga — DM 24,50
Quarterback (Festplatten-Backup) — DM 119,–

HK-Computer

F. Hansmann & Th. Küpper GbR
Bonner Straße 37 ** 5000 Köln 1

RAM-Erweiterung AMIGA 500 auf 1MB — DM 198,–

mit Uhr, abschaltbar
dto. Platine mit Uhr & Schalter ohne RAMs — DM 79,–
dto. Leerplatine mit Stecker — DM 39,–

RAM-Erweiterung II AMIGA 500 auf 1MB — DM 239,–

Megabittechnologie, mit Uhr, abschaltbar
dto. Platine mit Uhr & Schalter ohne RAMs — DM 79,–
dto. Leerplatine mit Stecker — DM 35,–

AMIGA-Bremse * der Highscore-Killer * — DM 39,50

– regelt die Geschwindigkeit stufenlos bis zum Stillstand
– ideal für schnelle Games und Bildschirmfotografie

BOOT-Selector für Amiga — DM 14,50

wahlweise Booten von DF0: oder DF1: oder DF2: oder DF3:
bei Bestellung bitte angeben

Drive-Expander — DM 39,–

– für externe Laufwerke ohne Busdurchführung, einstellbare Laufwerksnummer, keine Kabellängenprobleme, abschaltbar
bei Verwendung eines beliebigen Boot-Selectors kann von jedem Laufwerk gebootet werden

Maus & Joystick-Adapter — DM 44,50

– für gleichzeitigen Anschluß von Maus und Joystick, mit LED-Anzeige, alle Maussteuerleitungen elektronisch geschaltet

Modem Set Discovery 1200C + — DM 298,–

– inkl. AMIGA-Anschlußkabel und DFÜ-Software

Modem Set Discovery 2400C — DM 448,–

– inkl. AMIGA-Anschlußkabel und DFÜ-Software
beide Modems sind Exportmodelle ohne Postzulassung! Inbetriebnahme strafbar

BTX/VTX Decoder mit FTZ-Zulassung — DM 248,–

Trackdisplay A2000 intern DF0: & DF1: — DM 98,–

(neuer Power-LED Träger mit integrierten 7-Segmentanzeigen)

Trackdisplay extern DF0: bis DF3: — DM 79,–

für jedes Laufwerk einstellbar

Kick-ROM — DM 49,–

- Kickstartumschaltplatine für zwei OriginalROMs
OriginalROM 1.2 oder 1.3 — DM 65,–
Kick-ROM mit einem OriginalROM — DM 98,–
KickstartEproms 1.2/1.3/Guardian — DM 99,–

Drucker, Farbbänder, Kabel, Staubschutzhauben und Software auf Anfrage

Telefon: 0221/311606 * Telefax: 0221/321166
Mo-Fr. 10.00-13.30 u. 14.30-18.30, Sa. 10.00-14.00
Stadtsparkasse Köln, Kto 6342133, BLZ 37050198

»PowerFire« Das Superding! — DM 29,50

Dauerfeuerinterface für Joystick und Maus
– optimale Impulsfolge für jedes Game einstellbar
– Dauerfeuer wird über Feuer- bzw. Maustaste aktiviert
– einfach zwischen Maus/Joystick und Rechner stecken
– abschaltbar

Neu ab Dezember • Neu ab Dezember • Neu ab Dezember

RAM-Erweiterung AMIGA 500 auf 2,3MB — DM 698,–

mit Uhr, abschaltbar, intern!
dto. Platine mit Uhr&Schalter ohne RAMs — DM 198,–

RAM-Erweiterung AMIGA 2000 — DM 848,–

8MB mit 2MB bestückt, autokonfigurierend, einfach durch zusätzliche RAMs und Jumper umstecken aufrüstbar, keine neuen PALs erforderlich
dto. Platine teilbestückt ohne RAMs — DM 498,–
RAM-Satz für 2 Megabyte — DM 498,–

Kickstartumschaltplatine 3-fach — DM 59,–

– für zwei OriginalROMs und eine Epromversion
Umschaltplatine mit einem Epromsatz — DM 155,–
Umschaltplatine mit einem OriginalROM — DM 108,–

Genlock-Interface — DM 248,–

nur Computer/nur Video/beide Bilder gemischt durchgeführter Monitorport,
Positiv/Negativ
externes Gehäuse

Professional View

Video-Digitizer der neuen Generation — DM 998,–
On Board RGB-Splitter bereits integriert !!
On Board Genlock-Interface bereits integriert !!
Realtime-Digitalisierung in 16 Graustufen, 25 Bilder/s
HAM-Modus mit 4096 Farben
kurze Realtime Animationen in s/w, Übertragung 3,5MB/s !
Alle Bilder und Animationen auf Diskette speicherbar
für Animationen mind. 1MB RAM
unterstützt 1MB ChipRAM

Neu ab Dezember • Neu ab Dezember • Neu ab Dezember

Haben Sie Hard- oder Software für den Amiga entwickelt ? Wir bieten Ihnen eine großzügige Umsatzprovision und eine ehrliche Abrechnung.

»»»Sprechen Sie uns an«««

Wir reparieren Ihren Amiga und Zubehör schnell und preisgünstig.

UPS-Versand: Nachnahme 10 DM, Vorauskasse 5 DM, Großgeräte nach UPS-Tabelle ohne Aufschlag, Ausland nur gegen Vorauskasse + 15DM

Fordern Sie unser kostenloses Info an

Händleranfragen erwünscht

Gewußt wie!

Die Seiten für Einsteiger

Auch diesmal wollen wir Ihnen, die Sie gerade Ihren neuen Amiga auf den Schreibtisch gestellt haben, Hilfestellung bei anstehenden Problemen geben. Probleme aller Art – ob mit BASIC oder Assembler, Modula, C oder dem CLI – hier finden Sie Hinweise und Tips zu anstehenden Problemen.

Das "Editor-Problem"

Vielen wird bereits aufgefallen sein, daß einige Programme entscheidende Nachteile aufgrund ihrer mangelhaften Editoren haben. Zwar sind der SEKA-Assembler oder AmigaBASIC durchaus leistungsfähige Programme, aber durch den schlechten Editor, der dem Anwender als erstes auffällt, treten ihre Vorteile in den Hintergrund. Um diesen Nachteil auszugleichen, kann man die Multitasking-Fähigkeit des Amiga ausnutzen: Man läßt einfach einen namhaften Editor (z.B. KickED) gleichzeitig laufen und kann so die Vorteile beider Programme nutzen, indem man das im Editor erstellte Programm speichert und mit dem SEKA oder AmigaBASIC wieder lädt. Auf diese Weise wird automatisch verhindert, daß der gesamte Programmtext bei einem "Guru" verlorengeht, weil er noch nicht abgespeichert war.

Im Gegensatz zum Seka-Assembler muß man bei AmigaBASIC beachten, daß der Programmtext mit

```
SAVE"name",A
```

abgespeichert wird, damit er mit einem Editor weiterverarbeitet werden kann. Geladen wird er mit

```
NEW
MERGE"name"
```

im Direktmodus.

Blocks in AmigaBASIC

AmigaBASIC speichert Blocks, die mit "CUT" oder "COPY" erstellt wurden in der RAM-Disk (RAM:) unter dem Namen "BasicClip" ab. Dies ist ein ASCII-File, der aber anstelle des sonst üblichen Charakters 10 ($0a) den Charakter 13 ($0d) für ein Linefeed enthält. Nach einer kleinen Konvertierung dürfte sich jedoch jeder ASCII-File mit "PASTE" in einen bestehenden Programmtext einfügen lassen, sofern es als "RAM:BasicClip" in der RAM-Disk vorhanden ist. Dieses File wird zwar bei jeder Neudefinition eines Blockes überschrieben, bleibt jedoch auch nach Verlassen von AmigaBASIC bestehen. Hier ergeben sich für den geschickten Programmierer sicherlich einige Möglichkeiten.

Fehler im 68000?

Folgendes merkwürdige Phänomen wurde beim Arbeiten mit einem Assembler festgestellt:

Nach der Verknüpfung

```
AND.L A1,D1
```

also eines Adreßregisters mit einem Datenregister, ist eine Zuweisung wie

```
MOVEQ #7,D2
```

nicht mehr möglich. Dies äußert sich darin, daß das Programm zwar weiter abgearbeitet wird, aber ohne solche oder ähnliche Zuweisungen mittels des MOVE-Befehls wirklich auszuführen. Diese Erfahrung habe ich mit dem SEKA-Assembler gemacht, schließe jedoch aus, daß es an der Version desselben liegt, da er den Source-Code einwandfrei assembliert und disassembliert.

Nachlader

Im BASIC-Handbuch steht einiges zum Thema "Nachladen von Unterprogrammen in Maschinensprache".

Was dort aber nicht steht, ist, daß ein nachgeladenes Maschinencode-Programm unbedingt an einer geraden Adresse stehen muß. Will man sich die aufwendige Arbeit mit Betriebssystem-Routinen ersparen, so kann man es auf folgende Art und Weise tun:

```
  flen=300
laden:
  OPEN "Test" AS #1
  LEN=flen
  FIELD#1,flen AS a$
  GET#1,1
  programm$=a$
  einspr&=SADD(programm$)
  IF einspr&/2<>INT(einspr&/
2) THEN flen=flen+1:GOTO laden
```

(Bernd Rudolf)

Wie heißt der Befehl?

So schön der Command-Line-Interpreter für Direkteingaben ist, so schnell kommt man in die Verlegenheit, den einen oder anderen Befehl nicht mehr richtig zusammenzubekommen. Wie ging das noch mit PATH? Mußte jetzt erst der Weg zum Unter...? Nein, erst mußte der Parameter 1 eingegeben werden! Oder doch nicht?

CLI bietet uns für Fälle von überfallartigen Computer-Blackouts eine Hilfe, an die viele Amiga-Besitzer gar nicht oder nur wenig denken: Werden CLI-Befehle mit einem Leerzeichen und einem anschließenden Fragezeichen eingegeben, so gibt uns der Interpreter die jeweilige Syntax zum Befehl aus. Probieren wir es mit dem Befehl PATH:

Eingabe:

```
1> PATH ?
```

Ausgabe:

```
,,,,,,,,,,ADD/S,SHOW/S,RE
SET/S:
```

Danach bleibt der Cursor hinter der Ausgabezeile stehen und erwartet unsere neue Eingabe, diesmal mit korrekter Syntax.

Was bedeutet aber nun die Ausgabezeile? Die insgesamt zehn Kommas trennen einzelne Directoryeinträge, auf die wir den Suchpfad legen wollen. ADD übernimmt Directories in den Suchpfad, SHOW zeigt die aktuellen Pfade an, RESET löscht alle Pfadverzeichnisse außer dem SYS:C-Verzeichnis. Versuchen Sie es auch einmal mit 'LIST ?'.

Was bedeuten nun aber die Buchstaben A,K und S, die sich hinter der Befehlssyntax, getrennt durch einen Schrägstrich, befinden?

'/A' bedeutet, daß diese Option angegeben werden muß, '/K' erwartet im Anschluß noch Parameterwerte, und /S kann für sich alleine stehen. Übrigens: Versuchen Sie es doch mal mit INFO.

'Guten Tag' vom CLI

Das 'SAY'-Programm läßt sich auch vom CLI aus nutzen – wenn man einige Bedingungen anerkennt. Im CLI ist eine Direkteingabe zwar auch möglich

```
1>SAY "Hallo"
```

oder

```
1>SAY HALLO
```

und endet mit einem fröhlichen Hallo vom Rechner. Richtig interessant wird der Einsatz aber erst durch die Startup-Sequence und Dateien, die als Inhalt Sätze zum Sprechen haben.

Wird in der Startup-Sequence ein SAY-Befehl eingebaut, lassen sich gut Stellen markieren, an denen man sich zum Beispiel mit einem 'Guten Tag' begrüßen lassen kann.

SAY kann, wie bekannt, mit Dateien arbeiten; dazu gibt es die Option - X. Doch welche Dateien kommen nun in Frage? Befehle aus dem CLI wohl kaum, es sei denn, man gibt sich mit einigen unverständlichen Tönen zufrieden.

Die Lösung ist einfach: Mittels ED können wir eine ASCII-Datei erstellen, in dem die Sätze stehen, die unsere kleine 'Labertasche' vorbringen soll. Doch Vorsicht! Sätze wie "Hallo Amiga-User" bilden bei der Ausgabe Kauderwelsch, da der Amiga nur Phonemlaute übersetzen kann. Aus "Hallo Amiga User" muß also ein 'ha lo a mee ga user' werden. Probieren Sie doch mal, was so alles geht!

Mit 'Pipe' Zeit sparen

Wissen Sie, wie man mit einem Befehl drei Dateien löscht? Ganz einfach:

```
DELETE Datei1|Datei2|Datei3
```

Wir haben nichts weiter getan, als die drei Dateien mit dem PIPE-Symbol (zu erreichen durch gleichzeitiges Drücken von SHIFT und der Taste '\', die unter 'F10' zu finden ist) nacheinander zu löschen. Der jeweilige CLI-Befehl (das PIPE-Symbol kann von vielen Befehlen verwendet werden) sucht dann die Dateien heraus und beginnt mit der Bearbeitung jeder einzelnen. Benutzer eines Amiga mit Workbench kleiner als 1.3 müssen allerdings damit rechnen, daß das Pipe-

Symbol nicht einwandfrei funktioniert, da hier der Disk-Pufferspeicher benutzt wird. Sind mehr angegeben, werden die fünf zuerst gefundenen Dateien genommen; danach wird der Befehl beendet. Besitzer einer Kickstart ab Version 1.3 können dagegen die Zeilen vollschreiben.
Nützlich ist diese Befehlsart dann, wenn man zum Beispiel mehrere Dateien kopieren muß, aber den dazugehörigen Befehl nur einmal aufrufen will oder kann.

Wie groß ist die RAM-Disk?

Ob man einen Amiga 500 mit 512 kByte oder einen Amiga 2000 mit vier oder mehr MByte besitzt, überall findet man ein Laufwerk, das auf den ersten Blick gar nicht vorhanden zu sein scheint: die RAM-Disk. Sie ist eigentlich ein Phänomen, existiert sie doch gar nicht so, wie wir es von normalen Laufwerken gewöhnt sind: drehend, kratzend, eine Leuchtdiode zum Blitzen animierend. Und doch kann sie wichtige Funktionen übernehmen, nämlich ähnliche wie ein 'richtiges' Laufwerk. Was ist nun eigentlich die RAM-Disk? Sie wird aus dem Hauptspeicher des Amiga, dem RAM gebildet, wie der Name schon sagt. Da der Schreib- und Lesespeicher unterschiedliche Größen haben kann, zum Beispiel durch Speichererweiterungen, kann auch die Größe der RAM-Disk variieren.
Seltsamerweise bekommt man im CLI keine richtige Auskunft darüber, wie voll die RAM-Disk ist. Nach jedem Aufruf von INFO bekommt man gesagt, daß die RAM-Disk zu 100% belegt ist, dabei ist es unerheblich, ob sie 1 kByte oder 100 kByte an Programmen enthält. Seltsam, nicht?
Dabei ist die Erklärung recht einfach: Der Amiga verwaltet die RAM-Disk zu jeder Zeit. Und so wird der RAM-Disk mehr Platz zugewiesen, wenn wir ihn benötigen. Brauchen wir wieder weniger an Speicherplatz für das logische Laufwerk, wird der nicht benutzte Teil wieder dem normalen Schreib- und Lesespeicher zugeordnet. Der Amiga kann dies, weil die im Speicher vorhandenen Programme 'relocatibel', d.h. frei verschiebbar sind. Die RAM-Disk existiert deshalb auch nicht 'an einem Stück'. Sie befindet sich in Häppchen aufgeteilt mitten im normalen Speicher. Deshalb wird uns vom INFO-Befehl einfach mitgeteilt, die RAM-Disk wäre zu jedem Zeitpunkt voll. Glauben Sie das jetzt immer noch?
Wie nutzt man nun die RAM-Disk am besten? Nun, wie wäre es als Ersatz-Workbench? Vor allem Besitzer eines A 500 mit nur einem Laufwerk sollten sich diesen Schritt überlegen. Zuvor jedoch noch etwas Wichtiges: Gerade beim A 500 in 'Rohversion', also ohne Speichererweiterung, ist die RAM-Disk nicht allzu groß. Hier muß die Kopie der Workbench (Sie arbeiten doch hoffentlich nur mit Kopien?) abgespeckt, sprich von einigen Programmen und Tools erleichtert werden. Zu den Löschkandidaten gehören die Utilities mit der CLOCK und dem NOTEPAD, sowie die FONTS. Am besten benutzt man zum Löschen den CLI, da von der Workbench manche Löschoperationen nicht unterstützt werden.
In die Startup-Sequence dieser 'Klein-WB' schreiben wir noch folgende Zeilen nach dem letzten Kommando:

```
COPY SYS: RAM: ALL QUIET
ASSIGN SYS: RAM:
ASSIGN C: RAM:C
ASSIGN S: RAM:S
ASSIGN L: RAM:L
ASSIGN DEVS: RAM:DEVS
ASSIGN LIBS: RAM:LIBS
CD RAM:
```

Mit dem ersten Befehl kopieren wir alle Dateien von der Startdiskette ins RAM, mit den ASSIGN-Zuweisungen veranlassen wir das Amiga-DOS, zukünftige Befehle oder Programme von der RAM-Disk ablaufen zu lassen, und der letzte Befehl erklärt die RAM-Disk zum aktuellen Laufwerk. Ab jetzt brauchen Sie die Diskette in DF0: nicht mehr. Auch Programme lassen sich so ganz hervorragend auf die RAM-Disk umlegen. Aber aufgepaßt, immer auf den Speicherplatz achten.

INFOrmation ist alles

Auf der Workbench befindet sich innerhalb des Menüpunktes 'Workbench' die Funktion INFO. Wenn Sie diese Funktion noch nicht gesehen haben, lohnt es sich, sie einmal anzuwählen. Doch so ohne weiteres geht dies nicht! Die INFO-Funktion ist dazu da, Informationen über Programme oder Werkzeuge, die unter der Workbench zu gebrauchen sind, dem Benutzer anzuzeigen. Doch nicht nur das, man kann hier den Werkzeugen oder Programmen Parameter mit auf den Weg geben, die Veränderungen in dem jeweiligen Programm zulassen.

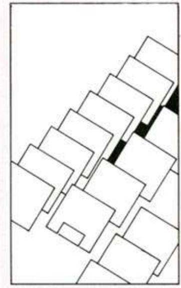
Einfachstes Beispiel ist der Editor 'Notepad'. Wenn man sein Icon anklickt, ärgert man sich immer wieder darüber, daß er nur als Winzling auf dem Monitor erscheint. Durch den Einsatz von INFO läßt sich dieses verbessern. Zuerst klicken wir das Notepad-Icon einmal an, so daß der Editor nicht gestartet wird. Danach gehen wir in die Menüleiste, wählen unter dem Titel 'Workbench' die Funktion INFO an. Nach kurzer Zeit bekommen wir die gewünschten Informationen über Notepad. Außer dem Namen des Programms stehen noch Infos über den benötigten Speicherplatz und die auf Diskette benötigten Blöcke. Im Stack-Feld läßt sich der benötigte Stapelspeicher erweitern, falls das Programm aufgrund zu kleinen Speichervorkommens abbricht (Vorsicht! Hier besser mit dem Handbuch arbeiten!) Während bei der Workbench 1.2 im Status-Feld nur DELETABLE/NOT DELETABLE einstellbar ist, also ob das File gelöscht oder nicht gelöscht werden kann, finden sich bei der Workbench 1.3 noch die Zusätze (READABLE /NOT READABLE, WRITEABLE/ NOT WRITEABLE und ARCHIVED/ NOT ARCHIVED. Außerdem finden wir ferner zwei Eingabefelder: das erste für Kommentare, das zweite für Parameter. Wenn wir jetzt das Feld ADD im Feld TOOL TYPES anklicken, können wir eine Eingabe machen. Diese sieht nun folgendermaßen aus:

```
WINDOW=000,000,640,256
```

Nach dem Anklicken von SAVE wird dieser Zusatzparameter zu 'Notepad' abgespeichert. Rufen wir jetzt Notepad noch einmal auf, benutzt der Editor den gesamten Bildschirm.

INFO bekommt damit plötzlich einen ganz anderen Stellenwert, oder?

(jb)



Tausche Tip gegen DATABOX

'Wir haben alle mal klein angefangen!' – Hinter dieser Überschrift verbirgt sich im Grunde nichts anderes, als Verständnis für die aufzubringen, die ihren Amiga erst seit kurzer Zeit haben und nun vor unüberwindlichen Problemen stehen, die so manchem Profi nur ein müdes Lächeln abringen.

Deshalb unser Aufruf an Fortgeschrittene: Wenn Sie einen guten Tip oder einen Griff in die Trickkiste wissen, lassen Sie von sich hören. Dabei ist es egal, ob es um BASIC, CLI, Assembler oder einfach nur Hinweise zum Arbeiten mit diesem Computer geht, nur eines ist Bedingung: Es muß interessant für Einsteiger sein, also nicht so: "Der COPY-Befehl wird zum Kopieren benutzt..." Sie sollen sich aber nicht ohne Anreiz an die Arbeit machen:

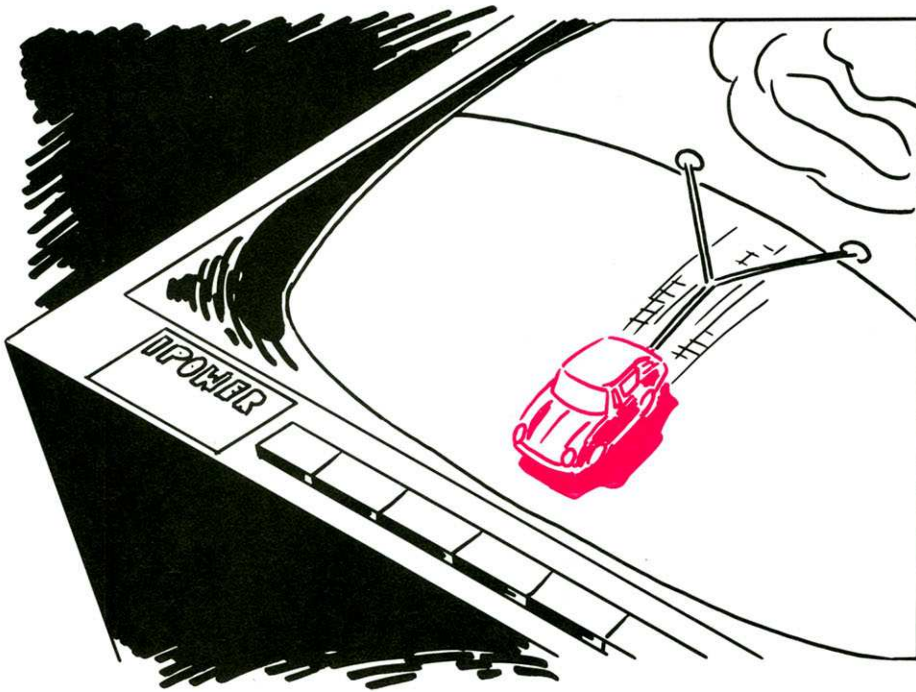
Wir bieten für jede Einsendung, die von uns für gut befunden und im Heft veröffentlicht wird, eine DATABOX der AMIGA DOS mit allen im Heft veröffentlichten Programmen.

Also ran an Tastatur und Maus und überlegt, was es zu verbessern oder zu erklären gibt. Alle Kurztips schicken Sie bitte als Textfile auf Diskette an:

Redaktion AMIGA DOS
Kennwort: Tausche Tip gegen DATABOX
DMV-Verlag
Postfach 250
3440 Eschwege

Ihre Diskette wird Ihnen nach Ansicht wieder zurückgeschickt, diejenigen, deren Tip angenommen wurde, finden gleichzeitig die neue DATABOX beiliegend.

Viel Spaß beim Tipsuchen, und denken Sie dran: Neueinsteiger werden es Ihnen danken.



Ulf Peterson

Die Blitter-Ecke

Hallo und herzlich willkommen zur neuen Blitter-Ecke von AMIGA DOS. Nicht, daß Sie jetzt denken, wir wiederholen nun zum x-ten Male nur die Dinge über den Blitter, die diverse Zeitschriften auch schon in etwas abgewandelter Form zu berichten wußten, denn in dem Fall könnten wir uns ja die Ecke sparen. Wir wollen vielmehr versuchen, so ausführlich und verständlich wie nur möglich Ihnen (fast) alle Geheimnisse des Blitters nahezubringen.

Mit anderen Worten heißt das, wir belassen es nicht nur bei der Erklärung, wie man Flächen kopiert, Linien zeichnet und Flächen ausfüllt, sondern wir werden Ihnen auch unter anderem zeigen, wie man BOBs erstellt oder Animationen erzeugt.

Besonders interessant dürfte für Sie dabei der BOBs-Artikel werden, denn unseres Wissens hat sich damit noch kein einziges Buch, geschweige denn eine Zeitschrift, beschäftigt. Wir haben uns also viel vorgenommen. Im übrigen werden wir in dieser Ecke zu keinem Zeitpunkt Gebrauch von den Routinen des Betriebssystems des Amiga machen, die für den Blitter zuständig sind. Wir könnten Ihnen dadurch zwar eine Menge Tipparbeit ersparen, jedoch haben Sie im Endeffekt davon sehr wenig. Wie Ihnen bekannt ist, arbeitet das Betriebssystem ja nicht gerade immer mit der erwünschten Höchstgeschwindigkeit. Aus diesem Grunde klammern wir das Operating System von vornherein aus, um den Lernanteil so hoch wie möglich zu gestalten. Um auch denjenigen den Anschluß an dieses Thema zu gewährleisten, die zwar Grundkenntnisse in Assembler besitzen, sich aber noch nicht ausführlich mit dem Amiga beschäftigt haben, wollen wir im ersten Teil erst einmal allgemein etwas über den Bildschirmaufbau und den Blitter sagen.

Der Bildschirm

Wie wohl allen bekannt sein dürfte, wird ein Fernseh- bzw. Monitorbild mit Hilfe des Elektronenstrahls zeilenweise aufgebaut. Alle 1/50stel Sekunden beginnt dieser Strahl immer wieder von neuem von der linken oberen Ecke des Bildschirms aus, das Bild Pixel für Pixel neu aufzubauen. Die zum Bildaufbau nötigen Informationen holt er sich dabei aus den sogenannten Bitplanes, wobei sich jeder auf dem Bildschirm sichtbare Pixel aus der Bitkombination der verwendeten Bitplanes errechnen läßt. Im Normalfall besitzen die Bitplanes alle dieselbe Größe, wie dies z.B. beim CLI oder der Workbench der Fall ist. Die Größe eines OS-Screens beträgt beim Lo-Res-Modus 320 x 256 Punkte. Da der Amiga allerdings mit dieser Einheit wenig zu tun hat, sagt man, daß der Bildschirm 40 Worte breit ist (40 x 16 = 320). Es ist wichtig, dies zu wissen, denn auch der Blitter "rechnet" in dieser Worteinheit. Da wir jetzt in kurzen Zügen unsere Kenntnisse über den Bildschirm erneut aufgefrischt haben, kommen wir jetzt zum Hauptthema, dem Blitter.

Der Blitter

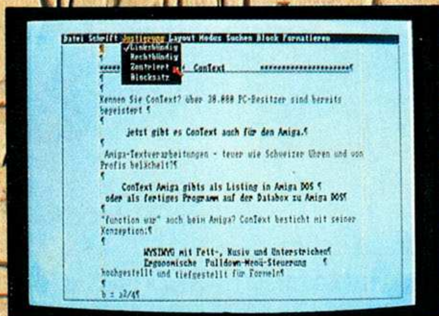
Wie der AGNUS ist auch der BLITTER ein Coprozessor des Amiga, das heißt, er nimmt der 68000er CPU zeitaufwendige Arbeit ab. Frei übersetzt bedeutet das Wort BLITTER (BLOCK IMAGE TRANSFERER) Datenblock-Bild-Kopierer. Ohne Zweifel macht er diesem Namen wirklich alle Ehre, denn das Verschieben von Datenblöcken geht mit einer Geschwindigkeit von bis zu 16 Millionen Punkten in der Sekunde vonstatten. Durch seine bereits oben beschriebenen Funktionen und der enormen Geschwindigkeit kommt der Blitter in fast allen grafischen Bereichen zum Einsatz, so unter anderem auch, wenn es gilt, in der Workbench ein Window zu verschieben. Es ist übrigens möglich, die Geschwindigkeit des BLITTERS geringfügig zu steigern, indem man ihm absolute Priorität über alle anderen Operationen gewährt. Dieses läßt sich durch Setzen von Bit 10 im DMACON Register erreichen. Zwar ist der BLITTER sehr schnell, allerdings kann man mit effizienter (das Gewünschte mit möglichst wenig Befehlen in möglichst kurzer Zeit erreichen) Programmierung erreichen, daß ein einfaches Programm Datenblöcke schneller als der Blitter verschiebt. Solche Routinen zu entwickeln, ist aber Sache von Profis. Kehren wir also zurück zum BLITTER. Wie oben bereits erwähnt, arbeitet auch der BLITTER mit Hilfe von Spalten und Zeilen. Es bereitet dem BLITTER also keine Mühe, beispielsweise eine komplette Bitplane im Bruchteil einer Sekunde zu kopieren. Allerdings wurden dem Blitter von den Entwicklern Grenzen gesetzt. So beträgt die maximale Anzahl an Worten, die der BLITTER kopieren kann, 65536 Worte, also den doppelten Adressierungsbereich eines 8-Bit-Computers wie dem C-64 oder ATARI XL/XE (128 kByte). Des weiteren kann die Größe eines BLITTER-Fensters horizontal maximal 64 Worte betragen und vertikal "nur" 1024 Zeilen. Somit wäre das BLITTER-Fenster mehr als sechsmal größer als ein normaler Lo-Res-Screen. Allerdings dürften nur wenige Programmierer von der Maximalgröße des BLITTER-Fensters Gebrauch machen.

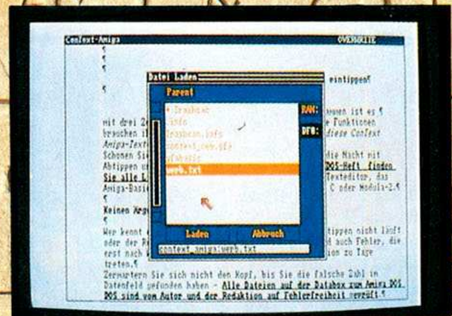
Damit wäre unser erster Einstieg in die Welt des BLITTERS getan. Wie gesagt, handelt es sich bei diesem Artikel nur um eine Einführung. Im nächsten Teil geht es dann das erste Mal wirklich zur Sache, wenn das Kopieren von Datenblöcken und das Thema Animation behandelt wird, dann wird auch das erste Listing präsentiert.

(mm)

Textverarbeitung im Wandel der Zeit

ConText AMIGA





Grabdenkmal im Besitz des Kestner-Müseums, Hannover

In der PC-Welt ist ConText bereits ein Begriff für Textverarbeitung:
Mehr als 30.000 Anwender sind von ConText PRO und ConText EASY begeistert, und jeden Tag werden es mehr.
Jetzt können Sie die einzigartige Konzeption von ConText auch auf dem Amiga erleben.
AMIGA DOS bringt ConText AMIGA als Listing zum Abtippen oder als fertiges Programm auf der Diskette über den Databox-Service.

Was ist am ConText-Konzept so einmalig?

Zuerst einmal das WYSIWYG-Prinzip: Sie sehen am Bildschirm Fettschrift auch wirklich fett und Kursivschrift als kursiv - ohne die sonst üblichen, zumeist unverständlichen Steuerzeichen.

Dann die doppelte Ansteuerung der Funktionen über eine mausbedienbare Menüleiste und über Tastenkombinationen: So kommen Sie immer auf dem schnellsten und bequemsten Weg zum Ziel.

Anwenderfreundliche und durchdachte Funktionen, die leicht zu bedienen und zu verstehen sind, machen Schluß mit dem "function war", der den Anwender mehr überfordert, als daß er ihm nutzt.

ConText AMIGA zum ersten

In der Ausgabe 1/90 von **AMIGA DOS** stellten wir Ihnen das Grundprogramm von **ConText AMIGA** als Listing mit folgenden Funktionen vor:

Datei:	neu, laden, speichern, quit, ConText-Format, Layout-Format, ASCII-Format
Schrift:	normal, fett, unterstrichen, kursiv, hoch, tief
Justierung:	linksbündig, rechtsbündig, zentriert, Blocksatz
Layout:	linker Rand, rechter Rand, Zeilenzahl
Modus:	insert, overwrite
Suchen:	suchen, ersetzen, weitersuchen
Block:	Start, Ende, kopieren, verschieben, löschen, freigeben
Formatieren:	Absatz

ConText AMIGA zum zweiten

Im Heft 2/90 von **AMIGA DOS** finden Sie, ebenfalls als Listing, ein separates Druckprogramm zu **ConText AMIGA**, mit dem Sie Ihre Texte entsprechend zu Papier bringen können.

Separat deshalb, damit **ConText AMIGA** für Sie im Hintergrund drucken kann, während Sie im Vordergrund weiter Ihrer Arbeit nachgehen können.

ConText AMIGA zum dritten

Natürlich bleibt es nicht dabei. – Wir arbeiten bereits an einer **ConText-AMIGA-Profi-Version**, die nicht nur die Möglichkeiten der PC-Version wie Breitschrift, Fußnotenverwaltung, Serienbriefe etc. umfassen wird, sondern diese noch übertreffen soll.

Neben eigenen Schriftfonts wird es auch möglich sein, Bilder in den Text zu integrieren und damit DTP-Dokumente zu erstellen. Bleiben Sie am Ball: **ConText AMIGA** – Ihr Programm für Textverarbeitung.

So bekomme ich ConText AMIGA

In **AMIGA DOS** 1/90 und 2/90 finden Sie die Listings in GFA-Basic Amiga.
Das Listing und das lauffähige, compilierte Programm finden Sie auf der Databox zu AMIGA DOS, ein Service, der alle Listings des Heftes auf Diskette anbietet.
Bitte benutzen Sie die Databox-Bestellkarte.

DMV-Verlag · Postfach 250 · 3440 Eschwege

DMV Daten- und Medienverlag

Die PD-Werkzeugkiste

Auch diesmal wollen wir wieder beweisen, daß PD-Programme kommerziellen Produkten kaum nachstehen. Besonderes Augenmerk haben wir diesmal auf die PD-Reihe CACTUS geworfen. Alle Programme dieser PD-Reihe sind ins Deutsche übersetzt und themenspezifisch geordnet worden.

Einer der mittlerweile immer größer werdenden Anzahl an PD-Clubs wartet mit dem etwas "wüstenzonischen" Namen Cactus auf. Ihrem Namen gerecht werdend, wählen sie auf ihren Disketteninhalten auch das Outfit dementsprechend. Als Einleitung tönt aus dem Lautsprecher des Amiga der digitalisierte Comedian-Harmonist-Song "Mein kleiner grüner Kaktus". Auch die Menüführung ist nicht von schlechten Eltern: die meisten Programme lassen sich bequem über die Workbench erreichen. Hierbei unterscheidet Cactus zwischen den eigentlichen Programmen und den (in deutsch gehaltenen) Anleitungen.

Textverarbeitungen auf dem Amiga – leider nur Raritäten?

Der Amiga-Benutzer lechzt geradezu nach brauchbaren Textverarbeitungsprogrammen. Leider ist der Amiganer auf diesem Gebiet alleingelassen. Von PCs oder dem Atari ST(E) in der Masse und Qualität der Textverarbeitungsprogramme längst geschlagen, trottet er seinen Kollegen weit hinterher, um sie vielleicht doch noch im Laufe der Zeit zu überholen.

Dieses Manko ist natürlich auch den im PD-Bereich aktiven Programmierern aufgefallen, die durch mehr oder minder qualitative Textverarbeitungen die große Lücke nun schließen wollen.

Das Programm **M.S.Text** teilt sich in drei große Menüs ein. Da wäre zuerst die Option Dokument, die sich mit der Diskettenverwaltung und der Drucktechnik befaßt. Hier können Textdateien eingeladen, gespeichert, gelöscht und angehängt (gemergt) werden. Zusätzlich verfügt das Menü über eine Druckereinstellung, in der vier verschiedene Schrifttypen (NLQ, PICA, ELITE und SCHMALSCHRIFT), der rechte und linke Rand, sowie die Zeilennumerierung eingestellt werden können.

Am Bildschirm wird die jeweilige Zeile und Spalte, in der sich der Cursor momentan befindet und der verbleibende Speicherplatz angezeigt.

Was wäre eine Textverarbeitung ohne Blockmenüs? Durch die Markierung von bestimmten Textabschnitten können die Blöcke dann gespeichert und an anderen Stellen des Textes eingefügt werden. Natürlich können auch Textabschnitte gelöscht oder ausgedruckt werden. Ein Manko zeigt jedoch die Markierung der Textabschnitte, die nur durch die momentane Cursorposition erreicht wird. Die Blockmarkierungen sollten eher mit der Maus durchgeführt werden können, da dies wesentlich komfortabler und problemloser vonstatten geht.

Ferner kann lediglich über Tastenbelegungen an den Textanfang und das Textende gesprungen werden. Dies war leider schon alles, was die Textverarbeitung zu bieten hat. Die verfügbaren Optionen reichen sicherlich nicht für eine komfortable Textverarbeitung aus. Hier fehlen einfach noch erforderliche Dinge, wie beispielsweise Grafikeinbindung, verschiedene Schrifttypen und eventuell auch ein Korrekturmodus, der sich aber aufgrund des geringen Speicherplatzes nur schwer realisieren läßt, wenn man bedenkt, daß M.S.Text nur 15 Seiten à 79 Zeilen verarbeiten kann.

Als kleines Bonbon befinden sich noch ein paar kleine Hilfsprogramme auf der Diskette. Da wäre beispielsweise das, mit dem man Programme, die sich normalerweise nur aus dem CLI aufrufen lassen, auch von der Workbench starten kann. Oder wie wäre es mit einem kleinen Texteditor mit kompletter Maussteuerung? Wer ASCII-Dateien formatiert oder mit Seitennumerierung ausdrucken lassen will, kommt auch nicht zu kurz.

(Jürgen Seibel/br)

Name: Cactus Nr. 25
Quelle: A.P.S. Electronic
Preis:
5,25-Zoll-Disk.: DM 1,50
3,5 Zoll Disk.: DM 2,80

Cactus Nummer 27: Iconmanipulationen leicht gemacht

Rund um die Iconbearbeitung geht es bei der Cactus-Diskette Numero 27.

Wie bei Textdateien ist es auch mit Icons möglich, Formate, die auf fremden Computern erstellt wurden, auf dem Amiga zu verarbeiten. Ein heikles Thema liegt jedoch bei der Kompatibilität der Icons auf den verschiedenen Rechnern. Das Programm **MultiView** stellt diese Tatsache unter Beweis. Hiermit lassen sich folgende Grafikformate einlesen:

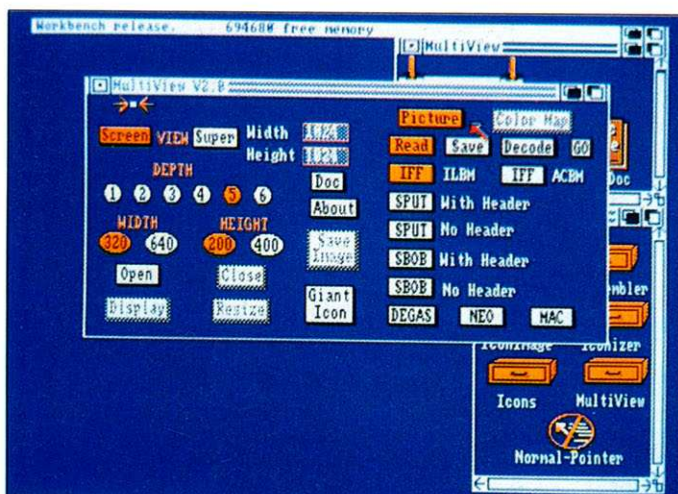
1. IFF – ILBM
   – ACBM
2. Atari ST – DEGAS
   – NEOchrome
3. Macintosh
4. SPUT – Sonderformat des BASIC-Befehls PUT
5. SBOB – Sonderformat der BASIC-BOBS
6. DECODE (dazu gleich mehr)

und die nachfolgenden Formate schreiben:

1. IFF – ILBM ungepackt
   – ACBM
2. SPUT
3. SBOB
4. Bildstruktur für C-Programmierer

Bei der Bildschirmeinstellung werden zwei Möglichkeiten, nämlich Screen und Superscreen, angeboten. Mit Screen wird ein normaler Amiga-Bildschirm, mit Superscreen dagegen ein überdimensionaler Bildschirm erzeugt, dessen maximale Pixelanzahl 1024*1024 beträgt. Kaum zwischen Screen und Superscreen entschieden, muß man schon eine zweite Wahl treffen. Doch keine Angst, Sie müssen ja nicht Miss World wählen, sondern lediglich die Anzahl der Farben, die natürlich auch für die Speicherkapazität von Relevanz ist. Je mehr Farben für das Bild genutzt werden, desto mehr CHIP-Speicher wird benötigt. Mit DECODE kann man sich Grafiken aus anderen Programmen (z.B. Spielen) in Detektivarbeit "herausholen". Ein kleines Beispiel auf der Diskette zeigt, wie's gemacht wird.

Sicherlich besitzen Sie einige Icons, die eher den Speicherplatz eines 64ers statt eines Amigas belästigen sollten. Besonders nachteilig ist es, daß von der Workbench bestimmte Icons (z.B. Drawer, Trashcan, oder Disk) voreingestellt sind. Bei diesen Symbolen ist es mit den üblichen Methoden der Workbench nicht möglich, sie zu manipulieren. Um dieser Misere ein Ende zu setzen, wurde eigens das Programm **IconImage** entwickelt, das es ermöglicht, Icons auszutauschen, ohne die Parameter des jeweiligen



Icon-Bearbeitung ist für den Anfänger keine einfache Geschichte: Mit der Cactus Nr. 27 soll sich dies ändern

Icons zu verändern. Der C-Source ist auf der Diskette enthalten.
Mit dem Pointer, dem kleinen Pfeil-Symbol, weckt man kaum Aufmerksamkeit bei Besuchern. Dieses Symbol etabliert sich lediglich in seinem Zweck, nämlich dem Aufrufen von Files innerhalb der Workbench-Umgebung. Um der Ästhetik und der Kreativität des Menschen einen Gefallen zu tun, läßt sich der Pointer aber auch modifizieren. Will man den Pointer jedoch öfter wechseln (warum auch immer), so muß der doch recht umständliche Umweg über die Preferences gegangen werden. Das Programm **Iconizer** umgeht dieses Problem. Hiermit läßt sich der derzeitige Mauszeiger unabhängig von den Preferences abspeichern und einladen. Die einzelnen Pointer werden in der Workbench-Umgebung als Icons angezeigt und können durch zweimaliges Anklicken aktiviert werden. Das zugehörige Programm **Icon2-Ptr** ist ein Hilfsprogramm, das lediglich zum Einladen der Pointer dient, und auch nur vom CLI aus gestartet werden kann.

Bildschirmgroße Icons – Der Icon-Assembler macht's möglich

Der **IconAssembler** übersetzt IFF-Brushes in Icons. Somit lassen sich auch übergroße, animierte Icons erstellen. Es wird sicherlich jedem Amiga-Benutzer klar sein, daß man mit einem Malprogramm wesentlich einfacher Icons erzeugen kann, als es mit dem IconEd der Fall ist. Mit Hilfe von IconAssembler können die IFF-Bilder dann als Icons abgespeichert werden. Hierbei muß jedoch erwähnt werden, daß das Programm den Icon-Type nicht verändert. Will man beispielsweise ein Disketten-Icon verändern, so muß auch ein Disketten-Icon angegeben werden, das manipuliert werden soll. Das Programm dient auch zum Modifizieren von Icons, das heißt aber nicht, daß Icons untereinander gemixt werden können. Dafür ist der IconEditor zuständig. Die Voraussetzung, daß die obig genannten Programme laufen, sind die Icons. So sind auf der Diskette Icons in die Bereiche Disk, Projekt, Drawer, Garbage und Tool gegliedert. Was geradezu provoziert, der Workbench einen neuen Anstrich zu geben.

(Jürgen Seibel/br)

Name: Cactus Nr. 27
Quelle: A.P.S. Electronic
Preis:
5,25 Zoll Disk.: DM 1,50
3,5 Zoll Disk.: DM 2,80



Rund um die Bild- und Soundbearbeitung geht es auf der Cactus Nr. 34. Mit dem Programm WBPic lassen sich IFF-Grafiken als Workbench-Hintergrund darstellen

Cactus Nr. 31 – Anwendungen en masse

Der Inhalt der Cactus-Diskette Nr. 31 widmet sich voll und ganz den Anwendungen.
Gute und ausführliche Programme haben einen großen Nachteil: Sie nehmen einfach zu viel Speicherplatz ein.
Über dieses Manko helfen sogenannte Packer hinweg, die Programme komprimieren (crunchen) können. Der **PowerPacker**, der sich auf dieser Diskette befindet, verfügt über einen Packer und einen Entcruncher. Mit dem Entcruncher werden komprimierte Programme in ihre ursprüngliche Gestalt zurückversetzt. Dabei können auch Programme, die von anderen Packern komprimiert wurden, entpackt werden. Warum entcrunchen? Nun, der Vorteil dabei liegt in der Überarbeitung bestimmter Programme, die in gecrunchter Version nur schwerlich bearbeitet werden können. Ferner kann das entcrunchte Programm auch wieder durch den Packer neu komprimiert werden, da es in der Verdichtungsfähigkeit der einzelnen Packer starke Unterschiede gibt. Beim Powerpacker ist es möglich, auch gecrunchte Programme ausführbar zu machen. Hierzu schreibt er eine "Entcrunch-Routine". Als äußerst positiv gestaltet sich das Script-Menü: Hier können bis zu 80 verschiedene Befehle aufgenommen werden, die nacheinander abgearbeitet werden. Dies ist ideal dafür wenn man mehrere Programme packen will.
Das Einlesen von Directories kann so manches mal zur Einschlafhilfe werden, da doch einige Zeit verstreicht, bis man das Diskettenverzeichnis auf dem Bildschirm hat. Auch hier kann diese Cactus-Diskette helfen: **FastDisk** soll Ihrem Laufwerk Beine machen. Aufgrund der Neuordnung und des Kopierens der einzelnen Sektoren reduziert FastDisk die Ladezeiten des Directories auf ein Fünftel. Zudem wird der Validierungsvorgang beschleunigt und die Ladezeit um 30 Prozent verringert. Einziger Nachteil: Das Programm setzt mindestens zwei Laufwerke voraus, wobei die Unterstützung der RAM-Disk nicht gegeben ist.
Das Programm **Tools** setzt sich aus drei kleineren Programmen zusammen:

– Ein Farbeinsteller, der aus dem CLI gestartet wird, verändert die Farben der Voreinstellungen, so daß beispielsweise eine neue, farbliche Darstellung der Workbench möglich ist. Damit Sie auch die Farbeinstellung Ihrer Träume verwirklichen können, steht Ihnen ein Testbild zur Verfügung.

– Die Stoppuhr mißt die Dauer von einzelnen Befehlsabläufen. So kann zum Beispiel ein Kopiervorgang, oder ein Ladevorgang gemessen werden.

– KeyMapEd erlaubt es, die im AmigaDOS mitgelieferten KeyMaps zu verändern, so daß Sie die einzelnen Tasten neu belegen können. Ihnen steht es nun frei, eine Taste mit acht verschiedenen Funktionen "vollzustopfen", oder gar das kyrillische Alphabet auf das Keyboard zu legen.

Das folgende Programm ist an unsere Hausfrauen oder Singles gerichtet, denen sich die undankbare Aufgabe stellt, mit wenig Geld den Haushalt in Ordnung zu halten. Solche Aufgaben erfordern natürlich eine präzise Buchführung, die das Computerprogramm **Haushaltsbuchführung** ermöglichen soll. Leider handelt es sich hierbei nur um die Demoversion des Programms. Die Folge ist, daß nur ein Dauerauftrag erteilt und ein Konto verwaltet werden kann. Dennoch ist ein Großteil der Optionen erhalten geblieben, zumal noch Übersichten von Gewinn bzw. Verlust und Jahresbilanzen, wie auch Kontenbilanzen, über die hauswirtschaftliche Lage Kunde geben. Zusätzlich unterstützt ein Taschenrechner die Auswertung der Bilanzen.
Mit dem letzten Programm, dem **Ansi-Editor**, lassen sich die Ansi-Steuerzeichen in Ihre Texte einfügen. Leider ist die Anleitung dieses Programmes nicht auf der Diskette enthalten.

(Jürgen Seibel/br)

Name: Cactus Nr. 31
Quelle: A.P.S. Electronic
Preis:
5,25 Zoll Disk.: DM 1,50
3,5 Zoll Disk.: DM 2,80

Cactus-Disk Nr. 34 – Hilfsprogramme zur Grafik-und Soundbearbeitung

Thema der CACTUS-Diskette Nr. 34 ist die Grafik- und Soundbearbeitung. Drei Programme zur Erstellung von Intros machen den Auftakt.
Da wäre als erstes das Programm **Amiga Message Center**, im folgenden kurz AMC genannt. Mit diesem Programm lassen sich problemlos Laufschriften erstellen,

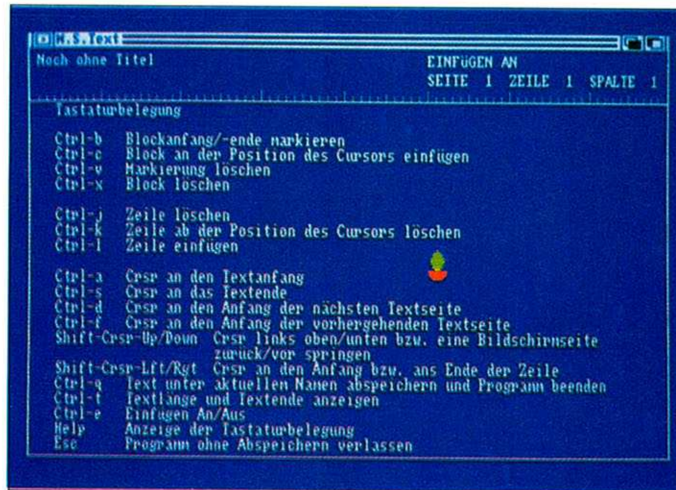
die einen Umfang von 1000 Zeichen beinhalten können. Um den Lauftext auf den Bildschirm zu bringen, muß der Text mit einem Editor (z.B. der Ed der Workbench) eingegeben werden. Dieser muß als ASCII-Datei vorliegen, und es dürfen nur Großbuchstaben verwendet worden sein (ASCII-Code 65 bis 90). Weiterhin können die Codes 32 bis 64 (Leerzeichen, Ausrufezeichen, Anführungsstriche etc...) verwandt werden. Hat man solch eine Textdatei erstellt, kann man sie in den AMC einladen. Der AMC macht aus der Textdatei nun ein eigenständig lauffähiges Intro, das unter einem beliebigen Namen abgespeichert werden kann.

Das Programm **Boot Intro** erzeugt, wie der Name schon sagt, auf einer Diskette ein Intro, das beim Booten der Diskette erscheint. Das Intro besteht aus einer Kopfzeile, deren Titel bis zu 44 Zeichen umfassen kann und einer Laufschrift, die aus maximal 300 Zeichen bestehen kann.

Bei Boot Intro kann direkt aus dem Programm der Text eingegeben werden, also braucht keine Textdatei eingeladen werden, wie es bei dem Programm AMC der Fall ist. Zudem besteht die Möglichkeit, sich das Intro vorab einmal anzusehen, ohne daß es abgespeichert werden muß. Neben einer individuellen Einschaltmeldung hat dieses Programm noch einen nützlichen Vorteil: Da der Bootblock für das Intro genutzt wird, kann man leicht feststellen, wann sich ein Bootblock-Virus auf die Diskette schreibt. Sollte nach dem Einschalten der mit Boot-Intro modifizierten Diskette die Laufschrift nicht mehr erscheinen, so ist die Möglichkeit gegeben, daß ein Virus zugeschlagen hat.

Rainbow-Writer heißt der dritte Intro-Maker im Bunde. Hier kann man sich einen beliebigen Text erstellen, der vor einem animierten Hintergrund abläuft. Bei diesem Programm kann der gesamte Bildschirm für einen (farblich gestalteten) Text genutzt werden.

Im Verzeichnis Bildwerkzeuge der Cactus 34 befinden sich äußerst fähige Grafik-Hilfsprogramme, die vom CLI aus aufgerufen werden können. Mit dem Befehl **IFF-EX** ist es möglich, Bilder von beliebiger Auflösung in ein ausführbares Programm umzuwandeln. Es benötigt also kein Ladeprogramm (z.B. Showpic), sondern kann durch einfachen Aufruf aus dem CLI gestartet werden. Zudem läßt es sich mit einem Packer komprimieren, so daß man umfangreiche Diashows erstellen kann. **Mod2Ex** ist ein Befehl, mit dem SoundTracker-Module als direkt ausführbare Programme abgespeichert werden. Somit wird keine Player-Routine für das Abspielen der Module benötigt. Der Befehl **IFF-Mod** ist eine Zusammensetzung von IFF-EX und Mod2Ex. Leider lassen sich Module, die mit der SoundTracker-Version 2.3 erstellt wurden, nicht implementieren. Bei Eingabe des Programmnamens wird dann das Bild angezeigt und gleichzeitig das Soundmodul abgespielt. Auch dieses Programm kann anschließend gepackt werden.



In Sachen Textverarbeitung sieht es auf dem Amiga äußerst traurig aus. M.S.Text auf der Cactus Nr. 25 ist ein kleines Programm, das für den Hausgebrauch durchaus ausreicht

Bounceshow ist ein Programm, mit dem sich kinderleicht Diashows erzeugen lassen können. Dazu muß man einfach eine Textdatei erzeugen, in der die Sounds und Bilder aufgerufen werden. Weiterhin muß in dieser Datei, die übrigens wie eine Startup-Sequence erstellt werden kann, der Zeitraum (wie lange das Bild gezeigt werden soll) angegeben werden. Hierbei kann man wählen, ob das jeweilige Bild per Mausdruck oder automatisch nach einer bestimmten Zeiteinheit gewechselt werden soll. Hat man sich solch ein Textfile erstellt, kann es mit einem Zusatzbefehl aufgerufen werden, und die Diashow ist perfekt. Das bzw. die Soundmodul(e) muß (müssen) übrigens im IFF-Format vorliegen.

3rd Day ist ein bedienungsfreundliches "Bilderklau-Programm" von knapp 12 kByte Länge. Der Amiga legt seine Grafiken im Chip-RAM ab. Wenn man einen Reset macht, wird der Speicher normalerweise nicht vollständig gelöscht.

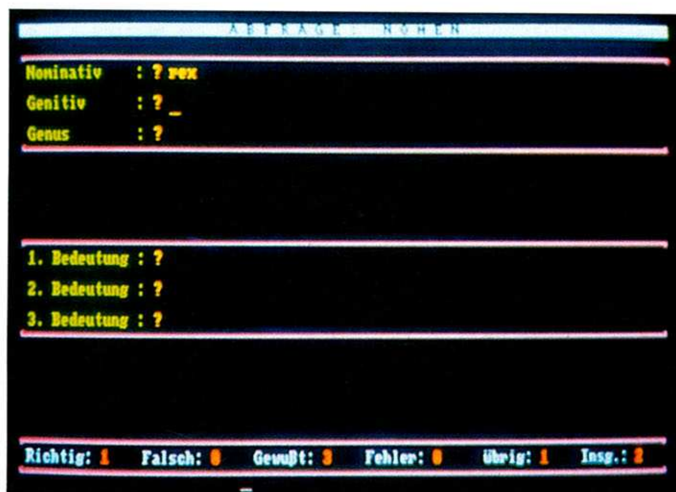
Aus diesem Grunde ist es möglich, die Bilder (sofern sie nicht gegen diese sogenannten Ripper geschützt sind) wieder aus dem Speicher herauszuholen. Dabei sucht 3rd Day nach Copperbefehlen, und den Bitplanes. Hat man ein Bild gefunden, so besteht die Möglichkeit es auf Diskette im IFF- oder RAW-Format abzuspeichern.

ST-ModulPlayer ist ein Programm, das Soundtracker-Musiken auf einem grafischen Hintergrund abspielt. Ein Musikstück von Jean Michel Jarre ist schon auf der Diskette als Demo vorhanden.

Last but not least befindet sich noch ein kleines Programm namens **WBPic** auf dieser Diskette, womit sich der Workbench-Hintergrund durch ein IFF-Bild ersetzen läßt. Dabei wird die Farbe 0 (im Normalfall ist das blau) durch die Grafik ersetzt. Die Bilder können durch einfaches Anklicken des jeweiligen Icons auf den Bildschirm geholt werden. Die Diskette enthält bereits neun Beispiele.

(br)

Name: Cactus Nr. 34
Quelle: A.P.S. Electronic
Preis:
5,25 Zoll Disk.: DM 1,50
3,5 Zoll Disk.: DM 2,80



Haben Sie Probleme mit lateinischen Vokabeln? Auf der RW 4 befindet sich ein Programm, das dem ein Ende setzen soll

RW 4 – Vom Lernunwilligen zum Sprachgenie

Für den eingefleischten Schulmuffel ist das Thema des Vokabelpaukens eine sicherlich leidige Angelegenheit. Wenn dann auch noch das Grundwissen zu oberflächlich herüberkam, bildet sich im Laufe der Zeit proportional zur wachsenden Vokabelmenge eine fulminante Lücke des Unwissens. Spätestens nach den Klausuren folgt dann die Ernüchterung des Schülers und das Gezeter der Erziehungsberechtigten. Eine Reise in das Ausland soll ja die Sprach-Barriere überwinden, doch wie wird das Problem bei einer toten Sprache wie Latein gelöst?

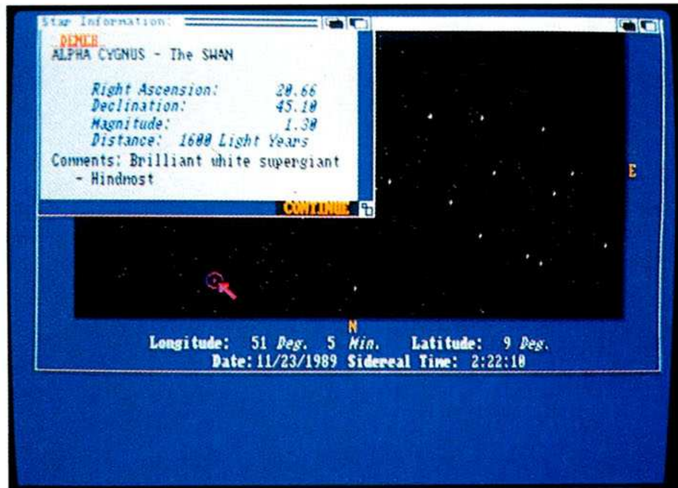
In diesem Fall soll der Vokabeltrainer auf der RW-Diskette Nr. 4 Abhilfe schaffen. Durch die Eingabe der Vokabeln baut sich ein Vokabelfeld nach folgenden Kriterien auf:

- Nomen (mit Genitiv, Genus und drei verschiedenen Bedeutungen)
- Verben (drei Stammformen und der Infinitiv)
- Adjektive (masculinum, femininum, neutrum)
- Adverbien
- Deponens (beide Stammformen und der Infinitiv)
- Semideponens (wie Deponens)

Diese Vokabeln werden natürlich in einer Datei abgespeichert, die auch ausgedruckt werden kann. Doch der Lerneffekt ist nicht die Eingabe der Vokabeln, sondern die Abfrage, die nach eigen gesetzten Parametern erfolgt. So kann der Rechner Deutsch-Latein oder Latein-Deutsch abfragen. Dabei werden die richtig und die falsch beantworteten Vokabeln nach Belieben mehrmals abgefragt.

(Jürgen Seibel/br)

Name: RW Nr. 4
Quelle: Wolf Computertechnik
Preis: ca. 6, – DM



Mit der Oase Nr. 44 können Sie die Sterne beobachten, auch wenn der Himmel einmal bedeckt ist

Der Sternenhimmel unter der Lupe – Star Chart

Star Chart ist ein Programm, das es erlaubt, etwa 600 Sterne, Galaxien und Nebel, die sich in der nördlichen Hemisphäre befinden, darzustellen und zu identifizieren. Dabei kann der Sternenhimmel für jeden geografischen Ort, jeden Tag, jede Zeit und jede Blickrichtung auf den Bildschirm gebracht werden. Diese Daten lassen sich zudem auf Diskette abspeichern. Um den Namen eines bestimmten Sternes (oder Sternbildes) und die Daten abzufragen, muß der Mauszeiger lediglich an die gewünschte Stelle auf dem "Monitor-Himmel" gebracht, und die Maustaste gedrückt werden. Die Daten, die bei der Abfrage angezeigt werden, sind:

- die allgemeine Bezeichnung
- der lateinische Name
- das Sternbild
- die Position
- die relative Helligkeit
- Entfernung zur Erde
- Anmerkungen zur Farbe und zum Namen, bzw. zur Galaxie oder anderen Sternobjekten.

Dabei wird der angewählte Stern auf der Karte mit einem Kreis gekennzeichnet. Es kann auch das gesamte Sternbild, zu dem der betreffende Stern zugehörig ist, angezeigt werden. Wer einen bestimmten Stern (oder ein Sternzeichen) sucht, dem kann auch geholfen werden. Er braucht nur in einem der vielen Hilfsmenüs den Namen einzugeben und schon wird dieser angezeigt. Eine auf dem Bildschirm angebrachte Uhr zeigt die jeweilige Zeit an, wobei die Zeiteinteilung auch manipuliert werden kann (Zeitraffer bis zu 30 Minuten pro Sekunde). Der einzige Nachteil den dieses Programm besitzt: Es ist nur unter AmigaDOS 1.2 lauffähig.

(br)

Name: Oase Nr. 44
Quelle: Wolf Computertechnik
Preis: ca. 10, – DM

Eine kleine Übersicht der Vertreiber von Public Domain, Free- und Shareware sowie Prüf-vor-Kauf-Programmen (ohne Anspruch auf Vollständigkeit)

A.P.S. Electronic
Sonnenborstel 31
3071 Steimbke
Tel.: 05026/1700

BAVARIANSOFT
Postfach 72
8473 Pfreimd

BELLINGRATH, CHRISTIAN
Hans-Böckler-Str. 55
5860 Iserlohn
Tel.: 02371/24192

DIGITAL MARKETING D. MÜCKTER
Krefelder Str. 16
5142 Hückelhoven-Baal
Tel.: 02435/2086,428, oder 1295

HIESKE, DIETER
Schillerstr. 36
6700 Ludwigshafen
Tel.: 0621/673105

KEIM, PETER
Vogelsanger Str. 34
5000 Köln 30
Tel.: 0221/520765

MAXON COMPUTER GMBH
Industriestr. 26
6236 Eschborn
Tel.: 06196/481811

Ossowski, Stefan
Veronikastr. 33
4300 Essen 1
Tel.: 0201/788778

TechnikSupport Verlag GmbH
Bundesallee 36-37
1000 Berlin 31
Tel.: 030/8621314

WOLF Computertechnik
Deipe Stegge 187
4420 Coesfeld
Tel.: 02541/2874

Österreich

KÜPPERS, BERND
Felberstr. 7
A-5730 Mittersill
Tel.: 06562/282

PETER RAUSCHERS COMPUTERSHOP
A-1100 Wien
Weldengasse 41
Tel.: 0222/621535

Public-Domain-Spieleshow

Ob Sie nun auf der Suche nach spannenden Abenteuern sind oder lieber ein rasantes Actionspiel riskieren, in dieser Ausgabe der Public-Domain-Spieleshow kommen Sie voll auf Ihre Kosten. Aber auch Freunde von Strategiespielen und Geschicklichkeitsproben werden bedient. Public Domain, das ist lebendige Amigaprogrammierung für aktive Anwender.

Castle of Doom

Im Sherwood Forest versteckt liegt eine alte Burg. Will Scarlet, ein einfacher Abenteurer, gerät auf seinen Reisen zu diesem scheinbar verlassenen Bauwerk. Ein alter Mann, der vor der Zugbrücke der Burg sitzt, erzählt Will von verborgenen Schätzen, die in den Mauern der Burg versteckt sein sollen.

Haben Sie ein wenig Zeit für ein Abenteuer? Wenn Sie diese Frage mit Ja beantworten können, dann sollten Sie vielleicht einmal einen Blick auf Castle werfen. Grundsätzlich handelt es sich hier um ein einfaches Textabenteuer. Der Spieler erhält alle Informationen über Orte und Schauplätze als Text. Wünscht der Spieler eine Aktion zu starten, so muß er diese in Form einfacher Zwei-Wort-Kommandos auf der Tastatur eintippen. Neben den erklärenden Texten wurden die meisten Örtlichkeiten, die der Spieler in der Burg besuchen kann, mit einem netten Bild versehen. Natürlich kann ein PD-Abenteuer wie Castle nicht mit kommerziellen Produkten dieses Spielgenres verglichen werden. Wer jedoch noch nie ein Abenteuer gespielt hat oder diese Art Spiel gerne mag, der sollte mit Castle einmal einen Versuch wagen.

Name: Castle of Doom
Enthalten auf: Cactus 29, Kickstart 152
Vertrieb: siehe Anbieterliste

Stoneage

Das Spielkonzept könnte kaum einfacher sein. Es gilt, in einem Höhlensystem Moosfetzen einzusammeln. Der kleine, grüne Moosfresser hat Hunger. Da das einfache Sammeln von Moos nun nicht gerade aufregend ist, gibt es da noch eine Menge Felsbrocken, die samt und sonders recht wackelig sitzen und beim geringsten Stoß wie eine Lawine herabstürzen. Wenn sich der kleine, grüne Moosfresser gerade darunter befindet, endet seine Mahlzeit fatal; der Stein zerquetscht ihn. Gelingt es allerdings, in einer Szene alle Moosfetzen innerhalb eines Zeitlimits aufzuessen, gelangt der Moosfresser in eine neue Szene, in der es neben vielen neuen Moosflecken auch eine ganze Menge gefährlicher Felsen gibt. Stoneage ist eher ein Denk- denn ein Reaktionsspiel.

Neben einer ganzen Reihe fertiger Spielszenen gehört zu Stoneage ein Editor, mit dem Levels nach eigenen Vorstellungen kreiert werden können.

Name: Stoneage
Enthalten auf: Kickstart 182
Vertrieb: siehe Anbieterliste

Blizzard

Kennen Sie ein Ballerspiel mit vielen bösen Gegnern, einem soft scrollenden Hintergrund, der sich aus gelungenen High-Tech-Grafiken zusammensetzt und das Public Domain ist? Wenn Sie nun glauben, das es so etwas nicht gibt, dann wird es höchste Zeit, daß Sie sich Blizzard einmal anschauen.

Der Spieler steuert einen Kampfgleiter, der über die Oberfläche einer Raumstation fliegt. Diese Station wird gerade durch eine Horde böser Außerirdischer angegriffen. Ihre Aufgabe ist es, ankommende Alienformationen mittels eines Lasergeschützes zu zerstören. Gelingt es, eine komplette Formation in die ewigen Jagdgründe zu schicken, winkt ein satter Extrabonus.

Blizzard ist ein echtes PD-Bonbon. Das horizontale Scrolling ist gekonnt programmiert, die Hintergründe bewegen sich sanft unter dem Gleiter dahin. Schafft der Spieler es, 10000 Punkte zu erreichen, winkt ein Besuch im zweiten Level, das sich sowohl durch seine Hintergrundgrafik, als auch durch die auftauchenden Aliens vom vorhergehenden unterscheidet.

Name: Blizzard
Enthalten auf: Kickstart PD 172
Vertrieb: siehe Anbieterliste

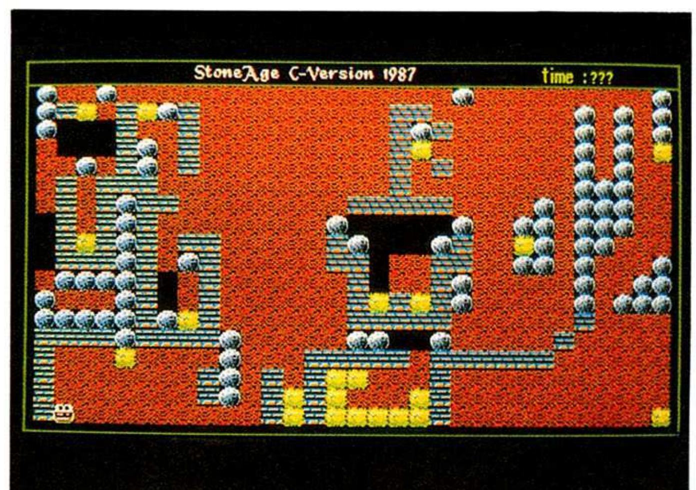
Kaiser II

Vom Leiter eines Handelskontors zum Reichskaiser – so könnte der Werdegang eines erfolgreichen Kaiserspielers aussehen. Kaiser II ist ein Gesellschaftsspiel mit leichten Parallelen zu Monopoly. Wie bereits erwähnt, ist das Ziel des Spielers die Kaiserkrone, die aber nur unter Einsatz von viel Geld und Gold erreicht werden kann. An einer Partie Kaiser können bis zu acht Spieler teilnehmen. Gespielt wird in Zügen – also ein Spieler nach dem anderen.

Jeder Spieler muß sowohl Handel treiben als auch dafür sorgen, daß sich die Infrastruktur seines Landes verbessert. Dies geschieht, in-



Betreten Sie die Burg des Unheils, und suchen Sie die versteckten Schätze



Moosfetzen müssen eingesammelt werden, aber herabfallende Felsen behindern Sie



Ein horizontal scrollendes Ballerspiel als Public Domain. Blizzard ist eine kleine Sensation



Wer Kaiser werden will, braucht viel Geld. Ein Gesellschaftsspiel, das mit mehreren Spielern besonders viel Spaß macht

dem Mühlen, Basare oder Lagerhäuser errichtet werden.

Ist man durch geschicktes Agieren erst einmal aus der ärgsten Geldnot heraus, besteht die Möglichkeit, eine Handelsflotte aufzubauen. Und sollte der Spieler dann wirklich zuviel Geld haben, dann kann er es sinnvoll in Form einer Kathedrale (kostet nur 371578 Goldstückchen) oder einer formschönen, wehrhaften Stadtmauer (ist schon für läppische 1181446 Goldstückchen zu haben) anlegen. Kaiser II ist ein Gesellschaftsspiel für lange Abende, das besonders mit mehreren Personen viel Spaß macht.

Name: Kaiser II
Enthalten auf: Bavarian Soft 115
Vertrieb: siehe Anbieterliste

Hayesinvader

Nehmen Sie Platz in der Hayes-Invader-Arena und machen Sie sich auf ein spannungsgeladenes Duell gegen eine erdrückende Überzahl von Angreifern gefaßt. Wenn Sie den Startcheck an den Instrumenten Ihres Kampfgleiters hinter sich gebracht haben, gibt es kein Zurück mehr. Sie stehen im Mittelpunkt einer riesigen Arena. Von allen Seiten nähern sich die unterschiedlichsten Vehikel, um über Sie herzufallen. Da kommen riesige Walker – mit Beinen versehene Panzer – auf sie zu. Panzer gleiten heran und feuern aus allen Rohren. Zielsuchraketen jagen Ihnen hinterher. Ihr Ziel in dieser Arena sind Punkte.

Hayesinvader präsentiert sich auf dem Monitor in animierter Vektorgrafik, was an sich für ein PD-Spiel schon eine Überraschung ist. Spielerisch erwartet Sie ein gnadenloses Duell, das einen ganz schön außer Atem bringen kann. Hayesinvaders ist für PD-Verhältnisse ein absolutes Superspiel, das ich unumwunden empfehlen kann.

Name: Hayesinvaders
Enthalten auf: Bavariasoft PD 98
Vertrieb: siehe Anbieterliste

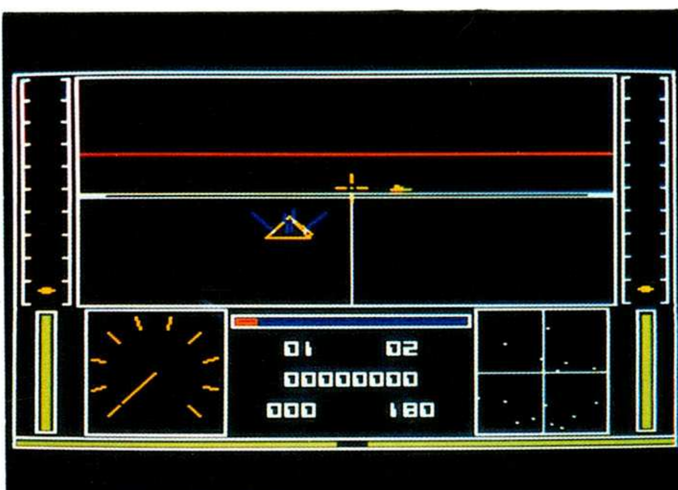
Peter's Quest

Daphne, Peters Freundin, ist entführt worden. Der findet die Entführung sehr unschön und macht sich auf, seine Liebste zu befreien. Der Weg führt ihn durch zwanzig verschiedene Bilder, an deren Ende die gefangene Daphne wartet. Unterwegs muß Peter Daphnes Spuren folgen, die in Form kleiner Herzchen in jedem Bild verstreut sind. Sind alle Herzen eingesammelt, geht es in der nächsten Szene weiter. Aber nicht nur Herzen muß Peter sammeln, er muß sich auch vor den Igeln vorsehen, die in einigen Bildern patrouillieren. Berührt Peter einen Igel, so ist ein Bildschirmleben verwirkt.

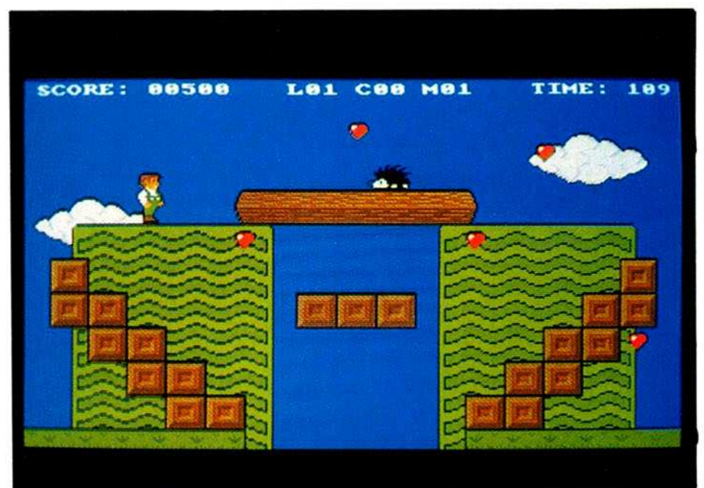
Wer Hüpf- und Sammelspiele mag, der liegt bei Peter's Quest genau richtig. Drei verschiedene Schwierigkeitsgrade und eine speicherbare High-Score-Tabelle ergänzen sich mit den gut gestalteten Grafiken und lustigen Soundeffekten, so daß aus Peter's Quest ein wirklich attraktives Spiel geworden ist.

Name: Peter's Quest
Enthalten auf: Oase PD 33
Vertrieb: siehe Anbieterliste

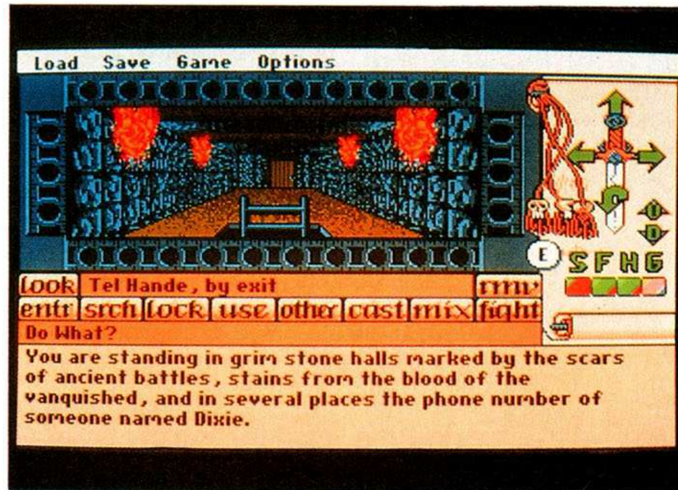
(hs)



3-D-Vectorgrafik vom Feinsten. In der Arena von Hayes Invaders ist die Hölle los



Daphne, Peters Liebling, ist entführt worden. Nur den Geschicktesten wird die Rettung gelingen



Keef, der Dieb, auf seinem Weg zu Reichtum und Macht



Oh Keef, wir waren mehr als großzügig mit dir, doch nun ist der Tag gekommen, an dem die Geduld der Meister ein Ende hat. So wie wir vom ersten Augenblick an, als wir dich als Baby fanden, wußten, daß du ein Leben auf der Straße führen wirst, so schicken wir dich nun dorthin.



Unsere Priester versuchten dich in den Künsten der Heilung zu unterweisen, doch du hattest nichts Besseres zu tun, als sie mit blasphemischen Reden zu verärgern. Dann versuchten unsere Krieger, einen Kämpfer aus dir zu machen. Doch auch dort versagtest du. Deine Mitschüler warfen dich vor die Türe, weil sie deinen dämlichen Tanz nach jedem Sieg nicht mehr sehen konnten. Dann versuchten unsere Zauberer, einen Magier aus dir zu machen. Doch du benutzt diese Kräfte, um Scherzartikel herzustellen.
Mein Sohn, du hast ein Problem.

Tja, Keef hat wirklich ein Problem, denn nun ist niemand mehr übrig, bei dem er in die Lehre gehen könnte. Doch Keef ist nicht auf den Kopf gefallen und schlägt eine Dieheslaufbahn ein. In der Stadt Merkon findet er ein lohnendes Betätigungsfeld. Hier gibt es Händler, Kneipen und Häuser, die man ausräubern kann. Tatsächlich sind die Anwohner vertrauensselig genug, die Türen offen und manchen wertvollen Gegenstand liegen zu lassen, aber sie sind auch gemein genug, ein paar hinterhältige Fallen zu installieren. Außer fiesen Fallen gilt es auch, der Stadtwache zu entgehen, die in regelmäßigen Abständen durch die Straßen patroulliert. Kommt die Wache an einem Haus vorbei, in dem Keef sich gerade befindet, endet dies mit einem Alarm, und Keef muß sich mit Waffengewalt seiner Haut wehren. Die gestohlenen Gegenstände, z.B. eine legendäre Zahnbürste oder ein Teekessel, der noch aus den dunklen Zeiten vor dem Imperium stammt, können in einer Kneipe bei einem Sammler in bare Münze verwandelt werden. Schon bald hat Keef die Stadt Tel Mercon gründlich abgesucht und alles geklaut, was nicht niet- und nagelfest war. Doch mit der Zeit kommt Langeweile auf. In einem Pub erfährt Keef von einem unterirdischen Verlies, das irgendwo südwestlich der Stadt liegen soll. Dort gäbe es noch einiges zu holen, meint der Krieger, der die Geschichte erzählt. Das interessiert Keef. Flugs besorgt er sich eine Karte des umliegenden Dschungels, und los gehts.

Rollenspiele präsentieren dem Spieler meist Spielstorys, in denen es darum geht, Welten zu retten, kosmischen Bösewichtern das Handwerk zu legen oder sonst irgendeine schicksalsträchtige Aufgabe zu erledigen. Keef ist da ein wenig anders. Der grundsätzliche Spielablauf ähnelt zwar dem konventionellen Rollenspiel und kommt nicht ohne Erfahrungspunkte, Kämpfe und Charakterwerte aus. Jedoch wird das, was man erleben kann, in einer Art und Weise präsentiert, die eher an eine Komödie erinnert denn an ein ernstes Rollenspiel.

Obwohl Keef grafisch eher mittelmäßig ausgefallen ist, hat mir das Spiel spontan gefallen. Dieses Programm ist ein Beispiel dafür, daß sich auch aus vielen mittelmäßigen Komponenten ein verdammt guter Cocktail mischen läßt – es kommt nur auf das Rezept an. Daß das Programm außerdem nicht kopiergeschützt ist, eine gut funktionierende Save-Option hat und sich sogar problemlos auf Festplatte installieren läßt, sorgt für eine recht unproblematische Handhabung und vermeidet unnötigen Ärger.

Keef the Thief ist kein Epos, das zwar einen Anfang aber kein Ende hat, vielmehr ist es ein Spiel, das Spaß machen soll und dies auch tut.

(hs)



Eine Kneipe ist immer ein Ort, an dem man Neuigkeiten erfahren kann

AMIGA DOS Blitzlicht

Name: Keef the Thief
Hersteller: Electronic Arts
Quelle: Fachhandel
Preis: 89,– DM

Positiv:
- humorvoll gestaltet
- flotter Spielablauf
- reeller Schwierigkeitsgrad

Negativ:
- hausbackene Grafik

AMIGA DOS URTEIL GUT

GRAFIK	SOUND	MOTIVATION
5	7	9



Karibik – wo andere Leute Urlaub machen, ballert Dyter, daß die Läufe glühen

Dyter 07

Die Karibik, der Zielort so manchen Traumurlaubs, scheint auch auf böse Außerirdische eine starke Anziehungskraft auszuüben. Sie haben sich auf einem zauberhaften kleinen Urlaubsatoll verschanzt und wollen nicht mehr weichen.

Natürlich muß die zauberhafte Südseeinsel zurückerobert werden. Ein besonders wirksames Werkzeug dafür ist schon gefunden worden – Dyter 07, ein schwerbewaffneter Kampfhubschrauber. Von einer Basis aus, die auf dem Atoll errichtet werden konnte, bricht Dyter zum Feldzug gegen die Aliens auf. Die haben bereits überall ihre Panzer und sonstigen Kriegsgerätschaften installiert und halten obendrein auch noch arme kleine Menschenwissenschaftler gefangen. Zerstört Dyter einen Panzer, kann es passieren, daß er damit auch einen Gelehrten befreit. Mit ein wenig Geschick kann dieser nun an Bord genommen werden. Gelingt es, den Wissenschaftler zurück zur Basis zu schaffen, wird er dort Extrawaffen für Dyter entwickeln.

Um die Aliens endgültig zu vertreiben, muß deren geheime Zentrale ausgeräuchert werden. Diese ist im Inneren eines Vulkans installiert, in den Dyter allerdings erst eindringen kann, wenn alle Roboter der Invasoren auf den Inseln des Atolls vernichtet sind. Einige dieser Roboter können nicht aus der Luft abgeschossen werden. Dann muß Dyter landen und einen Panzer absetzen, der in der Lage ist, die Gegner zu treffen. Neben den Extrawaffen vieler anderer Ballerspiele wirken Dyters Vierfachschuß, sein Flammenwerfer und die Lenkraketen eher possierlich. Aber erstaunlicherweise spielt sich Dyter hervorragend. Das Element, Extrawaffen erst einmal erarbeiten zu müssen, trägt viel zum Spielspaß bei.

(hs)

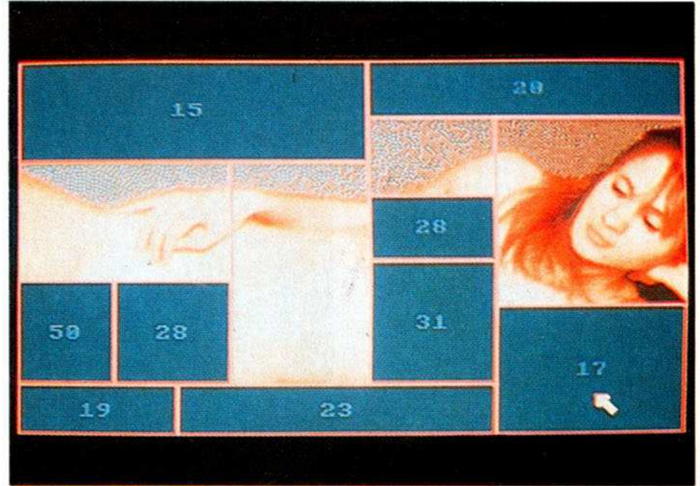
AMIGA DOS Blitzlicht

Name: Dyter 07
Hersteller: Reline
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- spannender Spielverlauf
- tolles Intro
- versteckte Extras im Spiel

Negativ:
- Steuerung gewöhnungsbedürftig





Mit den hübschen Mädchen entbrennt ein harter Kampf um jedes Stückchen nackte Haut

Centrefold Squares

Brettspiel-Umsetzungen sind ein beliebtes Thema für Programmierer. Da ist es kein Wunder, daß so gut wie jedes Brettspiel auch in irgendeiner Form auf dem Computer zu finden ist. Um die Spiele noch attraktiver zu gestalten, werden Extras zugefügt. So ist Centrefold Squares im Grunde nichts anderes als Reversi, das Sie gegen einige attraktive Damen spielen. Und wenn Sie gewinnen, dann können Sie die Mädels splitterfasernackt bewundern.

Um auch weniger geübten Spielern eine Chance zu geben, gibt es drei Gruppen von Damen, die mit sich spielen lassen. Ungeübte Spielerinnen, etwas versierte und echte Cracks. Jede der Damen verbirgt ihre Blöße zu Beginn der Partie hinter einer Sammlung von Rechtecken, die, strategisch postiert, nur das Gesicht der Schönheit freilassen. Jetzt wird um jedes Rechteck eine Runde Reversi gespielt. Verliert die Dame, verschwindet das Rechteck, um das gespielt wurde. Jedoch kann man die Rechtecke, um die man spielen möchte, nicht einfach aussuchen. Hier gilt die Regel: Jene Blöcke, die besonders uninteressante Körperpartien verdecken, sind zuerst dran. Um das Spiel aufzulockern, wurden einige Besonderheiten eingebaut: so werden vor Beginn des Spiels zufällig rote und grüne Steine verteilt, die, wenn sie besetzt werden, den Score um zehn Punkte verringern (rot) oder erhöhen (grün). Von Zeit zu Zeit erscheint ein gelbes Feld, das, wenn es besetzt wird, einen zweiten Zug erlaubt. Zusätzlich kann man, sollte sich das Blatt gegen einen wenden, sein Glück an einer Art Glücksspiel versuchen und auf diese Weise das Geschehen beeinflussen.

Da jedes Mädchen anders aussieht und spielt, ist für eine hohe Abwechslung und Motivation, vor allem für männliche Spieler, gesorgt.

(Robert Marz/hs)

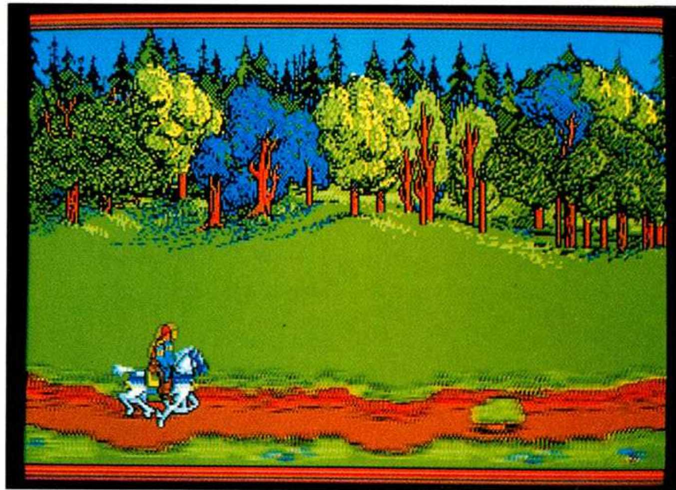
AMIGA DOS Blitzlicht

Name: Centrefold Sqares
Hersteller: CDS
Quelle: Fachhandel
Preis: ca. 70 DM

Positiv:
- neue Extras
- verschieden starke Gegner

Negativ:
- etwas eintöniger Spielverlauf





In der Stadt Hillsfar geht vieles nicht mit rechten Dingen zu. Kann Ihr Held alle Rätsel lösen?

Hillsfar

Am Ufer des Mondsees liegt die Stadt Hillsfar, die von Lord Maalthiir mit eiserner Hand regiert wird. Dorthin verschlägt es eine kleine Gruppe von Reisenden. Seit der Stadtrat gestürzt wurde, regiert Maalthiirs rote Garde unerbittlich. Ein unsicheres Pflaster für Abenteurer.

Das Abenteuer beginnt in einem kleinen Camp in der Nähe der Stadt. Zunächst gilt es, die Tücken des Reitens zu meistern. Der Charakter ist seit dem Moment seiner Entstehung mit einem Pferd ausgerüstet, mit dessen Hilfe Reisen über Land bewältigt werden. In dieser Sequenz ist der Reiter in der Seitenansicht zu sehen. Im Weg liegende Hindernisse müssen übersprungen werden. Gelingt dies nicht, besteht eine gewisse Gefahr, daß das Pferd das Weite sucht und der Abenteurer seinen Weg zu Fuß fortsetzen muß. In Hillsfar angekommen, empfiehlt es sich, zunächst die Stadt zu erkunden und den Stadtplan zu komplettieren. Danach ist ein Besuch in der entsprechenden Gilde von Bedeutung.

Kunst kommt von Können und Meister fallen nicht vom Himmel

Nach diesem Höflichkeitsbesuch steht zunächst ein Besuch bei Tannas auf dem Programm. Da in der Stadt niemand Waffen tragen darf, besteht nur hier eine Chance, die Skills im Umgang mit Waffen zu verbessern. Gegen eine kleine Gebühr kann unser Held nun wahlweise mit Sling, Darts, Dagger oder Pfeil und Bogen seine Künste an beweglichen und unbeweglichen Zielen erproben. Bei guten Schußleistungen können Geldpreise gewonnen werden. Bei besonders guten Plazierungen erfolgt dann der Hinweis, den Gildenmeister aufzusuchen. Neben einigen Erfahrungspunkten gibt es dort dann auch den ersten Auftrag, der von der Klasse und Rasse des Charakters abhängig ist.

Ruhm und Reichtum – aber bitte jetzt und sofort!

Der erste Auftrag führt auf den Friedhof, wo verlorengegangene Dokumente der Gilde wiederbeschafft werden müssen. Zu diesem Zweck muß unser Held nachts in das Friedhofsgebäude einbrechen. Das hört sich recht einfach an, die diversen Schlösser jedoch, die auffallende Ähnlichkeiten mit heutigen Schließanlagen aufweisen, sind nicht so einfach auszutricksen. Mit Hilfe eines magischen Knock-Ringes oder dem Zauberspruch 'Chime of Opening' öffnen sich die Schlösser aber wie von selbst. Der Chime of Opening kann übrigens im Old Wizards Labyrinth gefunden werden. Eine andere Methode besteht im Anheuern eines Diebes. Dieser bringt dann einen kompletten Satz Dietriche mit, mit denen man die zum Teil sehr komplizierten Schlösser knakken kann. Ist man erfolgreich eingebrochen, gilt es, das ganze Haus in kürzester Zeit zu durchsuchen. Herumstehende Kisten verbergen Schätze, Heiltränke, Knock-Ringe und gelegentlich auch die gesuchte Information oder einen wichtigen Hinweis. In manchen Gebäuden ist auch noch ein geheimer Treasure Room zu finden. Dieser befindet sich in der linken oberen Ecke des Gebäudes und kann entdeckt werden, wenn man die richtige Wandstelle berührt. Die dort zu findenden Schätze sind in der Regel natürlich wertvoller, als in den übrigen Räumen des Gebäudes. Nach einer gewissen Zeit öffnet sich dann ein Exit, durch den das Gebäude gefahrlos verlassen werden kann. Finden Sie den Exit nicht rechtzeitig, besteht die Gefahr, daß die Wachen Sie ergreifen. Entweder werden Sie um Ihr gefundenes Gold erleichtert und unsanft vor die Tür gesetzt, oder man verfrachtet Sie in die Arena, wo Sie um Ihr Leben kämpfen müssen. Diese Actionsequenz läßt sich jedoch recht gut meistern, wenn Sie vorher in den Pubs den Gesprächen der Bürger zugehört haben. Dort können Sie alle Hinweise bekommen, die notwendig sind, um selbst hartgesottene Monster besiegen zu können.

The Rod of Blasting oder Hilfsmittel erleichtern das Leben

Steigt man tiefer ins Spielgeschehen ein, stellt sich zunehmend das Problem der Überlandreisen. Nicht nur Hindernisse blockieren den Weg, sondern auch gelegentlich auftauchende Vögel versuchen, Sie mit einer Attacke aus dem Sattel zu heben. Das kann nicht nur das Pferd kosten, sondern auch eine erkleckliche Anzahl von Hitpoints. In dieser Situation kann der Rod of Blasting, ein Zauberstab, unschätzbare Dienste leisten. Er zerstört zuverlässig alle im Weg befindlichen Hindernisse, so daß unser Held nur den Vögeln auszuweichen braucht. Leider hat der Rod of Blasting nur eine begrenzte Anzahl von Ladungen und ist somit recht schnell verbraucht. Allerdings kann ein neuer Stab wieder im Mages Tower beschafft werden.

Deshalb ist es sinnvoll, bei erzielten Fortschritten den Weg ins Camp einzuschlagen. Nur dort kann der momentane Stand gespeichert werden. Sterben Sie dann bei einer Erkundung oder in der Arena, kann der letzte gespeicherte Spielstand wieder aufgerufen werden. Hillsfar ist in jeder Hinsicht ein bemerkenswertes Programm. Sowohl von der programmiertechnischen Seite als auch vom Aufbau der Rätsel wurde hier Maßarbeit geleistet. Der Spieler wird nicht durch endlose Kämpfe mit endlosen Monstergruppen von seiner Aufgabe abgehalten, sondern ist frei in seiner Entscheidung, auf welche Art und Weise und in welchem Zeitraum er die gestellte Aufgabe löst. Da ein äußerst weitläufiges Territorium erkundet werden muß, werden sicher einige Stunden intensiven Spielens vergehen, bis Hillsfar gelöst ist. Und die Aufgaben, die den Spieler erwarten, sind wahrlich nicht einfach.

(mm/hs)

AMIGA DOS Blitzlicht

Name: Hillsfar
Hersteller: SSI
Quelle: Fachhandel
Preis: 89,- DM

Positiv:
- komplett in Deutsch
- komplexes Spiel
- lange Spieldauer

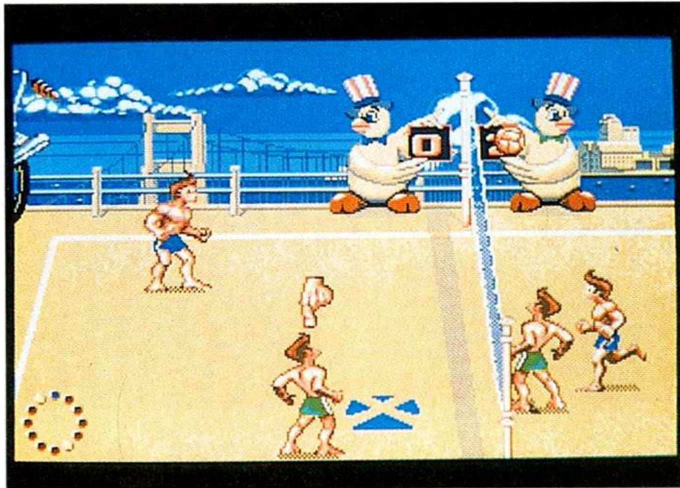
Negativ:
- nur PC Grafik





GRAFIK 6 · SOUND 3 · MOTIVATION 9





Volleyball vom Feinsten. Rund um die Welt warten gegnerische Teams auf Sie.

Beach Volley

Dies ist die märchenhafte Geschichte zweier Hinterhof-Volleyballer. Während die beiden bei einem heißen Match die Welt um sich vergessen, werden sie von einem Talentsucher beobachtet.

Und der verpflichtet die beiden zu einer Volleyball-Welttournee. Nun, Sie können in die Rolle eines dieser beiden Glückspilze schlüpfen und versuchen, für ihn die Spiele zu gewinnen.

Das erste Spiel müssen die beiden in London absolvieren, gelingt dies, geht es in so illustren Städten wie Sydney, Luxor, Nassau oder New York weiter. Jedoch haben die Götter vor das Vergnügen die Arbeit gesetzt, und dementsprechend sind bereits die ersten beiden Gegner zähe Spieler, die so manchen Trick kennen, der Punkte bringt.

Volleyball ist eigentlich ein Mannschaftssport, jedoch wird die Beach-Volley-Variante nur mit zwei Spielern pro Seite gespielt. Es gilt den Ball mit den Fäusten über ein Netz zu schlagen, das das Spielfeld in zwei Teile trennt. Von dort muß das gegnerische Team den Ball retournieren, ohne daß dieser den Boden berührt. Tut er es doch, gibt es einen Punkt für die andere Seite.

Zweifelsohne hat Ocean mit voller Absicht eine Volleyballversion mit nur zwei Spielern pro Mannschaft gewählt. Mehr Spieler auf dem Feld würden zu Lasten der Übersichtlichkeit gehen. Aber dafür wurde auf Grafik und Sound eine Extraportion Sorgfalt verwendet. Das Spiel beginnt mit einem comicartigen Intro, das von digitalisierten Soundeffekten untermalt wird. Das Spiel hält seinen Qualitätsstandard konsequent durch, und sei es nur eine Weltkarte, sie ist gut gezeichnet.

(hs)

AMIGA DOS Blitzlicht

Name: Beach Volley
Hersteller: Ocean
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- spannender Spielverlauf
- tolles Intro
- gute Grafik
- guter Sound

Negativ:
- Steuerung gewöhnungsbedürftig



Softwarenoten und wie sie zustandekommen

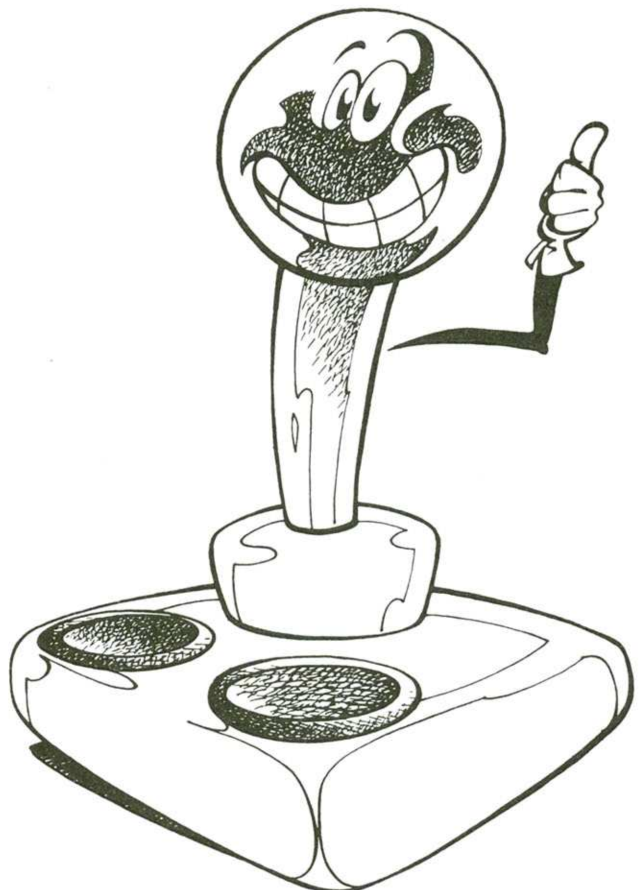
Daß Ihre AMIGA DOS bei den Spiele-Tests ein Urteil vergibt, haben Sie selbst sicherlich schon gesehen. Wie dieses Urteil funktioniert, das soll hier noch einmal im Detail erklärt werden.

AMIGA DOS beurteilt jedes Spiel anhand verschiedener Kriterien: Die Grafik, die Musik und den Spielspaß. Die Noten dafür werden in der kleinen Urteilsbox, die am Ende jeder Review zu finden ist, unter den Oberbegriffen Grafik, Sound und Motivation dargestellt. Die Note selbst richtet sich nach der Höhe des wiedergegebenen Wertes. Je höher, desto besser. Zu diesen Noten gesellt sich ferner ein individuelles Urteil des Testers. Eine vierte Note also, die allerdings nicht wiedergegeben wird.
Aus diesen vier Werten bildet sich schließlich das Gesamturteil:

- **Sehr gut**
- **gut**
- **o.k.**
- **geht so**
- **schlecht**



Die Bedeutung dieser Bewertungen erklärt sich von selbst. Uns bleibt noch, Ihnen viel Spaß mit dem AMIGA DOS Spieleteil zu wünschen.



THE MAN WITH THE HAT IS BACK.

AND THIS TIME HE'S BRINGING HIS DAD.

© UIP

INDIANA JONES and the Last Crusade

– Das Grafikadventure

Venedig, Österreich, und Berlin sind einige der Schauplätze, an die Sie das Indy-Grafikadventure entführt. Wenn Sie den Film verpaßt haben und das Actionspiel nicht mochten, dann wird dieses Spiel Ihnen bestimmt besser gefallen.

Sie erinnern sich an Zak McKracken, das erste Storyadventure aus dem Hause Lucasfilm? Hier konnten Sie in die Rolle eines Helden schlüpfen, der mit ein paar außerirdischen Invasoren haarsträubende Abenteuer erlebte. Nachdem dieses Spiel ein großer Erfolg wurde, nichts lag näher, als ein Abenteuerspiel mit einer so populären Figur wie Indiana Jones zu kreieren. Noch dazu ist Dr. Jones nun mal der Inbegriff eines Abenteurers.

Die Geschichte dreht sich um die Suche nach einem legendären Schatz, dem heiligen Gral, der der Legende nach schon von den Rittern der Tafelrunde gesucht wurde.

Indy begegnet dem Industriellen Walter Donovan, der an dem geheimnisvollen Behältnis, das einst das Blut Christi aufgefangen haben soll, großes Interesse zeigt. Er hat sogar Indys Vater, Dr. Henry Jones, engagiert. Und Henry Jones, seines Zeichens der größte Gralsgelehrte, wurde entführt. Indy macht sich auf die Suche nach seinem Dad und dem Gral. Das erste Ziel der Gralsjagd ist Venedig. Hier hielt sich Henry Jones zuletzt auf, und wie aus seinen Unterlagen hervorgeht, scheint er auch eine



Die Katakomben von Venedig verbergen noch manches Geheimnis vor Indy

wichtige Entdeckung gemacht zu haben. Nach einigen haarsträubenden Erlebnissen in den Katakomben unter der Stadt, entdeckt Indy das Grab eines Kreuzritters, auf dessen Schild eine Inschrift eingraviert ist. In diesem Text findet Indy Hinweise auf die Lage des Gralverstecks. Während seines Aufenthaltes in Venedig erfährt Indy auch, wo sein Vater steckt. Er wird auf einer Burg in Österreich gefangengehalten. Indy bricht nach Burg Brunwald auf, um seinen Daddy zu befreien.

Um Indiana Jones auf seiner Reise zu begleiten, zu unterstützen und zu steuern, bedarf es nur einer Maus. Die Steuerung erfolgt über das Anklicken von Kommandoworten, die am unteren Bildschirmrand positioniert sind. Hiermit können Sie Befehle wie nimm, benutze, ziehe oder drücke anwählen und mit Objekten und Personen aus dem darüberliegenden Bild verbinden und auslösen. Um in der Story voranzukommen, gilt es immer einige Aktionen zu erledigen. So muß Indy im Barnett College erst einmal seine Studenten beruhigen, das Päckchen mit dem Graltagebuch seines Vaters finden und sich mit Donovan treffen, bevor er nach Venedig weiterreisen kann. So kommt es, daß es bei diesem Spiel weniger auf gute Reaktionen, denn auf Köpfchen ankommt. Jedoch können auch jene mitunter von Vorteil sein. Damit ist gemeint, daß jedes Rätsel in diesem Spiel mit Köpfchen gelöst werden kann. Aber es gibt fast immer mehrere Wege ans Ziel zu kommen und meist ist auch einer dabei, der Indy bzw. dem Spieler einen harten Faustkampf beschert. So kommen Actionfreunde und Rätselcracks gleichermaßen auf ihre Kosten. Gute Grafik, passable Musik und Geräuscheffekte machen bei weitem nicht den ganzen Reiz dieses Spieles aus. Die Story ist es, aber auch die Rätsel, Fallen und Irrwege begeistern. Daß das Spiel nebenbei auch noch externe Laufwerke und Speichererweiterungen unterstützt, macht Indy, das Grafikadventure, nur noch spielbarer. Mein Eindruck – ein Spielgenuß erster Güte.

(hs)

AMIGA DOS Blitzlicht

Name: Indiana Jones and the Last Crusade
Hersteller: Lucasfilm
Quelle: Fachhandel
Preis: 89,– DM

Positiv:
- gute Story
- gute Grafik
- logische, durchdachte Rätsel
- kleine Actioneinlagen
- unterstützt Extraspeicher und Zweitlaufwerke

Negativ:
- wenig Musik und Geräusche





Die unheimliche Burg Brunwald. Hier wird Indys Vater gefangengehalten



Auf dem langen Weg zu seinem Erzfeind Joker muß Batman jetzt nur noch diesen Glockenturm erklimmen

Batman

Batman, der sympathische Fledermausheld, ist durch den gleichnamigen Film in letzter Zeit wieder in aller Munde. Damit man auch daheim den Kampf gegen den teuflischen Joker aufnehmen kann, hat Ocean eine Neuauflage des Batman-Spiels herausgebracht, die sich an den verschiedenen Szenen des Filmes orientiert.

Nachdem Batman erfahren hat, daß die Firma Axxis Chemicals von Jack Napier und seiner Bande überfallen werden soll, macht er sich natürlich sofort auf den Weg zu dem Gebäude. In der Fabrik muß er sich langsam zu Jack vorkämpfen, während Jacks Gehilfen aus Pistolen schießen und Handgranaten nach ihm werfen. Mit dem Batseil kann sich Batman von einer Etage zur nächsten hochziehen. Trifft er Jack, so fällt dieser in ein großes Faß mit giftiger Säure. Jack ist tot, aber der Joker lebt! Im zweiten Level muß Batman mit Vickie Vale und seinem Batmobil durch die Stadt fliehen und dabei aufpassen, daß ihn seine Verfolger nicht einholen. Auch gilt es, die Straßensperren der Polizei zu vermeiden, vor denen ihn ein Radar warnt. In der letzten Szene des Spiels kommt es dann zur endgültigen Auseinandersetzung zwischen Batman und Joker. Wer der Sieger sein wird, ist ungewiß...

Wer Batman mag, oder den Film besonders gelungen findet, der wird sich mit diesem Spiel gut amüsieren können.

Das Flair, das dem dunklen Ritter in den Comics anhaftet, bleibt allerdings vollkommen auf der Strecke.

Trotzdem ist Batman ein passables Actionspiel geworden. Sowohl die Klettersequenzen als auch die Fahrt mit dem Batmobil sind gut realisiert.

(hs)

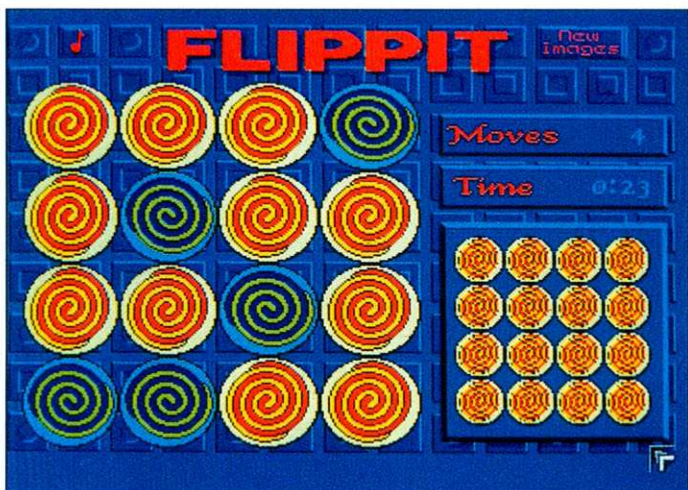
AMIGA DOS Blitzlicht

Name: Batman
Hersteller: OCEAN
Quelle: Fachhandel
Preis: 84,– DM

Positiv:
- gute Grafik und Animationen
- fünf unterschiedliche Level
- hohe Motivation, großer Spielspaß

Negativ:
- teilweise nervtötende Hintergrundmusik
- hoher Preis
- keine Passwörter





Die Steine müssen so gedreht werden, daß ein vorgegebenes Muster erscheint

Flipp It

Es gibt Glücksspiele, Spiele, die auf Geschicklichkeit basieren und Spiele, deren Ablauf strengen Regeln unterliegt. Gerade die präzise und logisch aufgebauten Spiele fordern vom Spieler wenig mehr als ein gutes Gedächtnis und genaue Beobachtung.

Die Regeln, die den Spielablauf von Flipp It bestimmen, hat jeder binnen Minuten begriffen, was aber nicht heißt, daß damit auch das Spiel gemeistert wäre.

Jedes Level setzt sich aus sechzehn Feldern zusammen, die in Form von vier mal vier Quadraten angeordnet sind. Jedes Feld hat zwei Seiten, von denen immer nur eine sichtbar ist und jede ein anderes Symbol trägt. Rechts neben dem Hauptspielfeld befindet sich ein kleines Window, in dem ein bestimmtes Muster dargestellt wird. Aufgabe des Spielers ist es nun, das Muster aus dem kleinen Fenster auf das große Spielfeld zu übertragen. Dazu müssen einzelne Steine mit der Maus angeklickt werden, die sich daraufhin drehen. Außer dem angeklickten Stein drehen sich aber auch immer einige andere Steine.

Dadurch wird das Herstellen des vorgegebenen Musters natürlich ungemein erschwert. Ist das Muster vollendet, werden die Züge und die Zeit erfaßt, die dazu benötigt wurden und in einer Highscoretabelle abgelegt.

Flipp It besteht aus einem Sortiment vorgefertigter Level und einer 'Randomizer'-Option, die neue Levels zufällig erzeugt.

Was an Flipp It gefällt, ist das Spielkonzept, das einfacher nicht sein könnte. Die technische Realisierung ist routiniert, aber ein wenig einfallslos geraten. Das Spiel selbst macht zwar viel Spaß, verliert allerdings mit der Zeit seinen Reiz.

(hs)

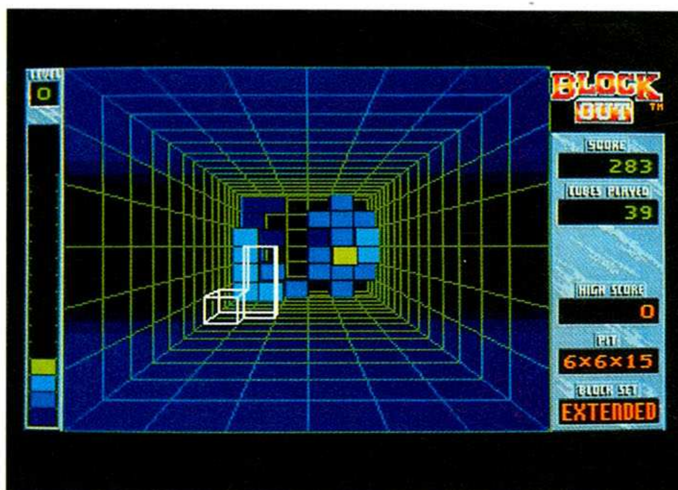
AMIGA DOS Blitzlicht

Name: Flipp It
Hersteller: Splash
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest.

Positiv:
- gutes Denksportspiel

Negativ:
- wird recht bald langweilig





Ein Schacht und viele Steine. Block out besteht aus wenigen Elementen und macht unglaublich viel Spaß!

Block out

Dieses Spiel ist ein Klon, daran ist nichts zu rütteln. Aber obwohl das Original schon ein echter Gassenhauer war, setzt Block out noch einen drauf. Hier müssen kompliziert geformte Steine zu geschlossenen Formationen gruppiert werden. Und zu allem Überfluß läuft hier alles dreidimensional ab.

Block out konfrontiert Sie mit einem Schacht, in den Sie direkt von oben hineinschauen. In diesen Schacht fallen unentwegt unterschiedlich geformte Blöcke. Diese Steine werden dreidimensional dargestellt und können vom Spieler über alle drei Raumachsen gedreht werden. Die Blöcke müssen auf dem Grund des Schachtes zu geschlossenen Schichten angeordnet werden. Gelingt es, die Steine so zu arrangieren, daß in einer Ebene keine Lücke bleibt, verschwindet diese Schicht.

Block out kann auf verschiedene Weise gespielt werden. Sowohl die Tiefe des Schachtes als auch dessen Abmessungen können in einem gewissen Rahmen modifiziert werden. Auch die Fallgeschwindigkeit der Blöcke ist einstellbar. Obwohl der Spielablauf von Block out im ersten Moment einen behäbigen Eindruck macht, kann eine Partie einen haarsträubenden Verlauf nehmen. Der Schwierigkeitsgrad eines Spieles ist auch von dem Steinsortiment abhängig, mit dem gerade gespielt wird.

Block out ist ein Geschicklichkeitsspiel, das unheimlich viel Spaß macht, auch wenn Sie mehr als einmal fluchen werden, weil ausgerechnet der Stein, den Sie dringend brauchen, einfach nicht erscheint (vor allem in den höheren Leveln).

(hs)

AMIGA DOS Blitzlicht

Name: Block out
Hersteller: California Dreams
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- fesselndes Spielkonzept
- verschiedene Spielmodi
- Spielkomponenten einstellbar

Negativ:
- bei größter Schachttiefe unübersichtlich
- Blockformen mitunter schwer erkennbar





Dragon Spirit versetzt Sie in den Körper eines magischen Drachens, dem bis zu drei Köpfe wachsen können. Wozu? Jeder Kopf ist in der Lage, einen vernichtenden Geschoßhagel abzufeuern

Dragon Spirit

Nur ein wahrlich tapferer Held kann jetzt noch die schöne Prinzessin Alicia befreien...

Drachen werden in der Regel mit finsteren, übelgelaunten Monstern in einen Topf geworfen. Daß dies lediglich ein Vorurteil ist, beweist Dragon Spirit. Denn Sie, der wackere Held dieses Actiongames, verwandeln sich in einen magischen Drachen, um den Gefahren trotzen zu können, die die Rettung der Prinzessin mit sich bringt. Insgesamt acht Zonen gilt es zu durchqueren, um zu dem Ort zu gelangen, an dem die Prinzessin gefangengehalten wird. Allerdings gibt es Unmengen von Monstern in diesem vertikal scrollenden Shoot'em Up, deren einzige Aufgabe darin besteht, Sie von diesem Vorhaben abzuhalten. Jedoch auch Ihre Bewaffnung hat so einige Glanzstücke aufzuweisen. Immerhin drei Köpfe können maximal erreicht werden, was auch die Feuerkraft, die aus Schüssen und Bomben besteht, recht ordentlich aufstockt. Gelegentlich herumliegende Eier bergen noch weitere Extras, wie Schilder, Homing Fire, Extralifes und vieles andere mehr. Allerdings bringt nicht jedes gefundene Icon auch Vorteile. Ein Totenkopf bewirkt, daß Sie einen Energieverlust in Form eines Kopfes hinnehmen müssen. Am Ende jeder Zone wartet dann ein Obermonster, das sich um so leichter niederringen läßt, je stärker Ihre Bewaffnung ist. Fünf Drachen stehen zu Beginn zur Verfügung, die allerdings schnell aufgebraucht sind, wenn man der Beschaffung von Extrawaffen keinen Vorrang einräumt. Alles in allem ein recht unterhaltsames Shoot'em Up, das seinen Reiz hauptsächlich aus der Verwendung eines Drachens als Kampfmittel bezieht.

(mm)

AMIGA DOS Blitzlicht

Name: Dragon Spirit
Hersteller: Domark
Quelle: Bomico Softwareteam
Preis: 69,95 DM

Positiv:
- verschiedene Szenarien
- zahlreiche Extrawaffen
- nette Spielidee
- steigender Schwierigkeitsgrad

Negativ:
- Geschwindigkeit gering



M2AMIGA MODULA-2

DAS MEISTVERKAUFTE, SCHNELLSTE, BEWÄHRTESTE UND KOMFORTABELSTE MODULA-2 SYSTEM AUF DEM AMIGA GIBT ES IN EINER NEUEN VERSION MIT VIELEN PRAKTISCHEN ERWEITERUNGEN.

COMPILER	SFR.	DM
M2AMIGA VERSION 3.3	270.00	342.00
DEMODISKETTE (FISH-DISK 113)	10.00	10.00
UPDATE-PREIS	10.00	10.00
(ZUM UPDATE ORIGINALDISKETTE EINSENDEN)		

ZU M2AMIGA IST EINE LANGE LISTE VON WERKZEUGEN UND BIBLIOTHEKEN ERHÄLTLICH:

WERKZEUGE	SFR.	DM
SOURCE-LEVEL-DEBUGGER	180.00	228.00
AUTOMATISCHES MAKE	80.00	108.30
M2APSE ENVIRONMENT	80.00	108.30
M2DECODER	80.00	108.30
OBJEKTCONVERTER	80.00	108.30
SOURCECODE + RTS	80.00	108.30
IFF-BIBLIOTHEK	80.00	108.30
SPEED-EDITOR	80.00	108.30

TREASURES-BIBLIOTHEKEN	SFR.	DM
AMIGA TREASURES	158.00	201.78
FILE TREASURES	158.00	201.78
MODULA TREASURES	78.00	102.60
MATH TREASURES	78.00	102.60
TREASURES-DEMODISKETTE	10.00	10.00

REPORT-BIBLIOTHEKEN	SFR.	DM
INTUITION REPORT	80.00	108.30
DEVICE REPORT	80.00	108.30
GRAPHIC REPORT	80.00	108.30
REPORT-DEMODISKETTE	10.00	10.00

PD-DISKETTEN	SFR.	DM
AMOK PD-DISK, JE	10.00	10.00
TREASURES-PD-DISKETTE, JE	10.00	10.00

DIE GENANNTEN PREISE SIND UNVERBINDLICH

DIE MODULA-2 LEUTE:

BUNDESREPUBLIK DEUTSCHLAND:
- H.J.MIELE-DATENTECHNIK GMBH 02983/8307
- SOS SOFTWARE SERVICE GMBH 0821/571081
- SW-DATENTECHNIK GMBH 04106/3998
- WILKEN & SABELBERG 0531/42689
- INTERPLAN-MUHLERT GMBH 089/1234066
- ADVANCED APPLICATIONS GMBH 0721/700912

SCHWEIZ:
- FREI-ELEKTRONIK 01/945 54 32

OESTERREICH:
- ICA ELEKTRONISCHE GERÄTE GMBH 0222/4545010

BEZUG AUCH BEI IHREM NÄCHSTEN COMPUTER- ODER BUCHHÄNDLER

GENERALVERTRIEB FÜR EUROPA:
A+L AG
DÄDERIZ 61
CH-2540 GRENCHEN/SO
TEL. (0041/0)65/52 03 11



Lufthockey ist ein Spiel, bei dem es auf schnelle Reaktionen ankommt. Haben Sie das Zeug zum Champion?

Shufflepuck Cafe

Als es noch keine Computerspiele gab, vergnügten sich die Besucher von Spielhallen mit Tischfußball, Flippern und Lufthockey.

Lufthockey wird von zwei Personen gegeneinander gespielt. Das Match findet auf einem Tisch statt, der rundum mit einer Bande versehen ist. Die Spieler stehen einander gegenüber und versuchen, einen runden Puck in das Tor des Gegners zu schießen. Der Tisch hat die Besonderheit, daß er auf dem Spielfeld über und über mit winzigen Löchern versehen ist, aus denen Luft strömt. Der flache Puck schwebt auf diesem Luftkissen und erreicht dank der verringerten Reibung, sobald er mit dem Schläger angestoßen wird, schnell eine hohe Geschwindigkeit. Der Spieler muß entsprechend gute Reaktionen aufweisen, wenn er verhindern will, daß der Gegner das eigene Tor trifft. Schufflepuck Cafe verwandelt aber nicht nur Ihren Amiga-Bildschirm in eine Lufthockeyarena, es stellt auch gleich eine ganze Auswahl unterschiedlicher Kontrahenten zur Verfügung, gegen die es zu bestehen gilt. Insgesamt gibt es neun Gegner, die neben ihrer unterschiedlichen Spielstärke auch noch verschieden aussehen und ganz individuelle Spielweisen an den Tag legen. Da gibt es den Echsenmann, der während des Matches ständig an seinem Sektkelch nippt und spätestens nach dem fünften Tor so betrunken ist, daß er seinen Schläger nicht mehr halten kann.

Shufflepuck Cafe ist sicher nichts, was man als Superhit bezeichnen könnte, dazu ist das, was geboten wird, ein wenig mager. Insbesondere der Umstand, daß nur gegen Computerspieler angetreten werden kann, ist ärgerlich. Trotzdem ist Shufflepuck Cafe ein Spiel, das auch auf Dauer immer wieder zu einem Match einlädt.

(hs)

AMIGA DOS Blitzlicht

Name: Shufflepuck Cafe
Hersteller: Broderbund
Quelle: Fachhandel
Preis: 79,95 DM

Positiv:
- neun verschiedene Gegner
- Option für Meisterschaften
- einstellbare Handicaps

Negativ:
- kann nur gegen den Computer gespielt werden.





Der Traum vieler Hobbystrategen besteht darin, Feldherr im Amerikanischen Bürgerkrieg zu sein. Mit North & South kann man sich einmal so richtig austoben und die Geschichte verändern

North & South

Der Amerikanische Bürgerkrieg war schon das Vorbild vieler Strategiespiele. Infogrames hat sich dieses Themas auf eine besondere Weise angenommen, die den sonst eher trockenen Spielen dieser Art einen neuen Touch verleiht.

Zum Angriff! Das Trompetensignal schallt über das Schlachtfeld, die Artillerie bezieht Stellung, die Kavallerie reitet auf Teufel komm raus, und die Infanterie macht sich zu einem Vorstoß bereit. Solche Szenen bekommt man bei North & South öfter zu sehen, vorausgesetzt, Sie spielen im Arcademodus. Im Titelbild kann man zwischen dem bereits erwähnten Arcademodus und einem Strategiemode wählen. Im Arcademode werden alle Schlachten per Joystick ausgefochten, während im Strategiemode der Computer automatisch den Ausgang einer Schlacht errechnet. Daneben können noch einige weitere Optionen eingestellt werden, z.B. ob es Stürme gibt, oder ob Indianer- und Mexikanerübergriffe stattfinden und letztendlich, ob die Parteien mit Unterstützung aus Europa rechnen können. Das Spiel selbst gestaltet sich ähnlich wie "Defender of the Crown", nur daß diesmal der Sezessionskrieg als Hintergrund dient. Der Spieler muß seine Armeen bewegen, Länder erobern und für Nachschub sorgen. Dabei helfen Eisenbahnlinien ungemein. Außerdem können hier Goldtransporte getätigt werden, für deren Erlös neue Armeen gekauft werden können. Viele lustige Animationen und Soundeffekte gestalten das Spiel zu einem echten Erlebnis.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: North & South
Hersteller: Infogrames
Quelle: Fachhandel
Preis: 84,95 DM

Positiv:
- sehr schöne Grafiken und Animationen
- gute Musik und Soundeffekte
- unterschiedliche Computergegner
- viele lustige Features
- ausführliche und witzige Anleitung

Negativ:
- kriegerische Handlung



AMIGA DOS – Spieletips

Heiße Tips zu harten Spielen – unter diesem Motto stehen auch in diesem Monat wieder unsere Helpline-Seiten. Und wahrlich, wir haben es uns nicht leicht gemacht. Für Sierra-Freunde gibt es einen ganzen Sack voller Tips zu Manhunter New York. Fantasyfans kommen bei den Tips zu Heroes of the Lance auf ihre Kosten. Natürlich haben wir auch ansonsten ganz tief in der Trickkiste gewühlt. Was dabei herauskam, lesen Sie auf den folgenden Seiten.

Die Suche nach dem verlorenen Juwel (Disk) von Mishakal hat schon mehr als einem Amiga-Spieler das Nervenkostüm ruiniert. Wie Sie in den Ruinen von Xak Tsaroth erfolgreich bestehen können, dazu möchten wir einige Tips geben.

Heroes of the Lance

In Xak Tsaroth werden Sie vielen verschiedenen Monstern begegnen. Einige von ihnen sind sehr mächtig und können für die Party zu einer echten Gefahr werden. Jedes Untier verfolgt im Kampf eine ganz bestimmte Taktik. Um gleich zu Beginn einer Konfrontation einen kleinen Vorteil zu haben, empfiehlt sich folgende Vorgehensweise. Postieren Sie den Kämpfer Raistlin Majere an einem der ersten vier Stellplätze. Taucht nun ein Gegner auf, können Sie die Pausefunktion aktivieren und einen WEB-Zauberspruch vorbereiten. Menschliche Gegner, wie z.B. Krieger, sind zwar gefährlich, aber berechenbar. Sie haben außer Ihrem Schwert keine weiteren Waffen, die Ihnen gefährlich werden könnten. Die in den Ruinen lebenden Trolle sind mit Vorsicht zu genießen. Sie sind gefährliche Kämpfer, nur sehr schwer zu verwunden, und sie sind ausgezeichnet geschützt. Hier hilft ein WEB-Zauber ungemein. Riesenspinnen sind eigentlich eher ärgerlich denn gefährlich. Sollten Sie jedoch gerade in einen Kampf mit irgendeinem anderen Ungeheuer verwickelt sein, und ausgerechnet in diesem Moment erscheinen die Spinnen, kann sich daraus eine heikle Situation für Ihre Kämpfer ergeben. Setzen Sie Flint an die erste Stelle der Party. Dadurch, daß er kleiner ist, kann er die Spinne auch noch bekämpfen, wenn sie schon ganz nahe herangekommen ist. Bozak Draconians können zaubern. Dadurch werden diese Monster auch auf weitere Distanz sehr gefährlich. Mit dem richtigen Zauberspruch – Web oder Sleep – auf den Lippen können aber auch diese Untiere bezwungen werden. Sollte das Zaubern nicht so klappen, dann können Sie auch den Zwerg Flint in die erste Reihe stellen. Er ist klein, so daß ihn die meisten magischen Angriffe verfehlen. Die gefährlichsten Bewohner Xak Tsaroths dürften mit Sicherheit die brütenden schwarzen Drachen sein. Die Biester lassen niemanden an sich heran, so daß ein Kampf mit Schwert oder Axt ausscheidet. Zuallererst sollten Sie die Charaktere mit dem Zauber 'Deflect Dragon

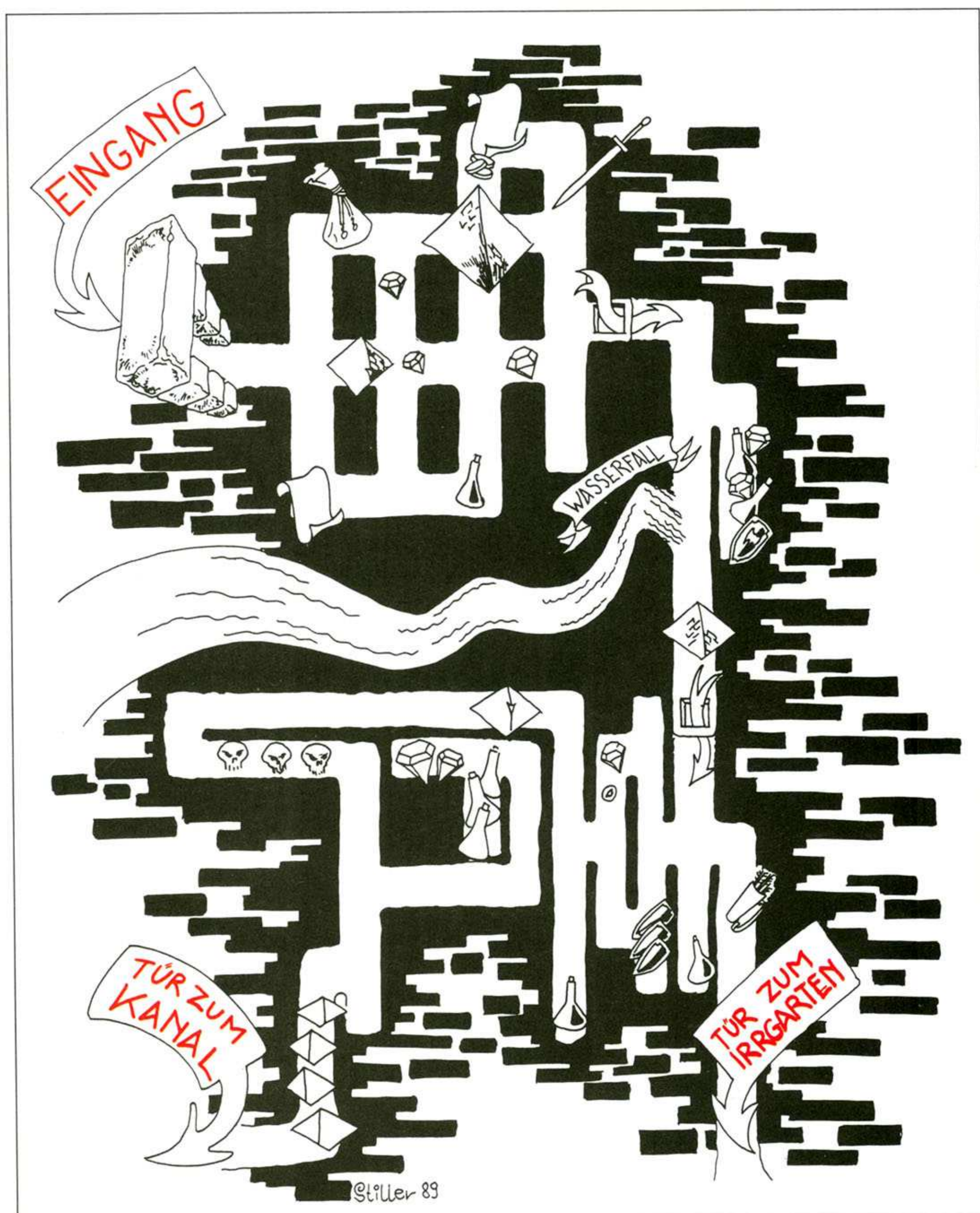


Bild 1: Der Eingang in die Stadt Xak Tsaroth

Breath' schützen. Mit einem Web-Zauber, der allerdings flach über dem Boden gezaubert werden muß, kann nun die Bewegungsfreiheit der Drachen so weit eingeschränkt werden, daß die Untiere auch mit einem Schwert bekämpft werden können.

Den genauen Weg durch Xak Tsaroth können Sie auf den Karten, die wir von dieser Stadt angefertigt haben, nachlesen. Denken Sie daran, daß nicht alle Gruben übersprungen werden können und manche nur von einem bestimmten Charakter. Um es nicht zu einfach zu machen, haben wir einige Schätze nicht eingezeichnet und bei den Gruben nicht angegeben, ob Sie übersprungen werden können oder nicht. Ein wenig Arbeit soll ja auch noch für Sie übrigbleiben. Infocoms Journey ist eine recht orginelle Variante des Abenteuers. Hier kann man schnell feststellen, daß auch ein Abenteuer ohne manuelle Texteingaben ganz schön kompliziert und schwierig sein kann.

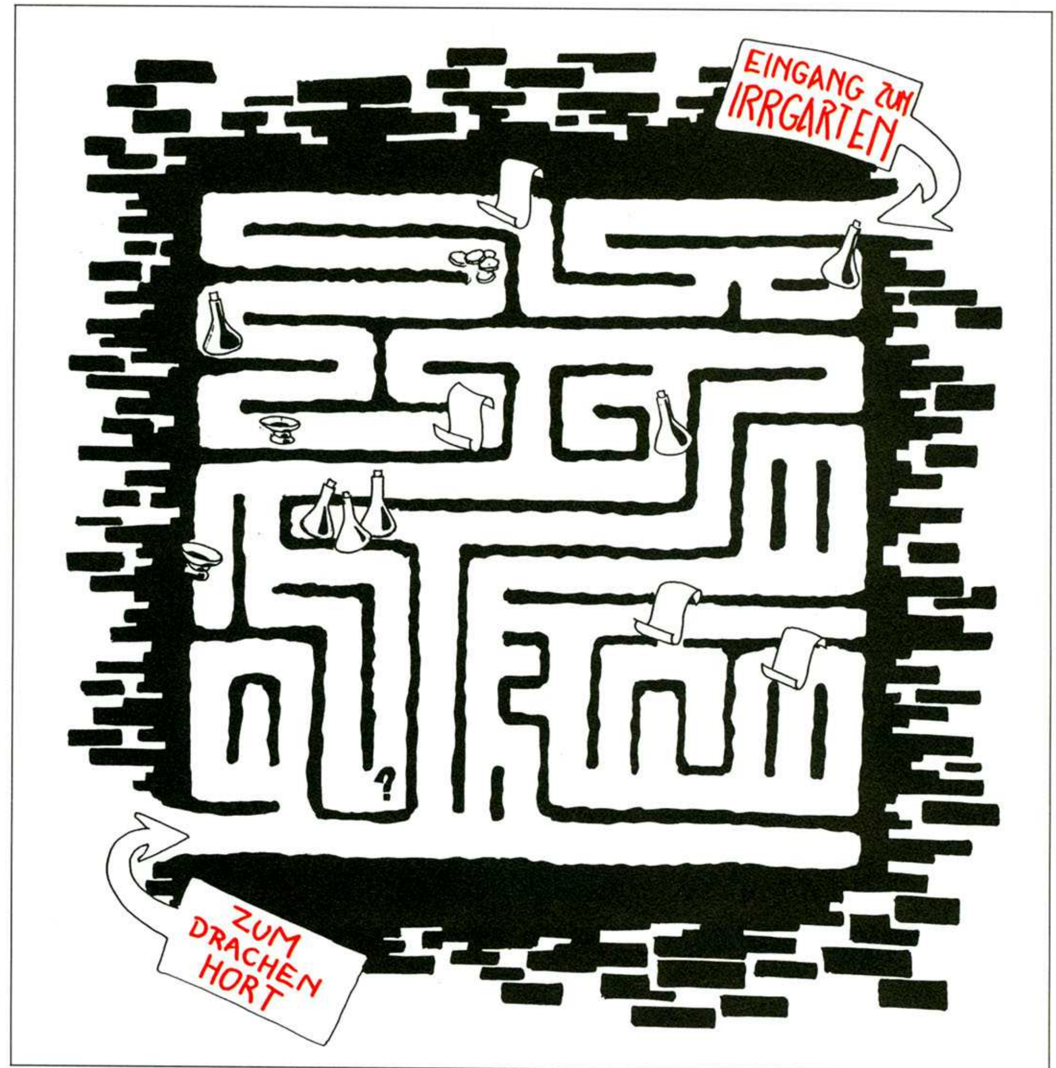


Bild 2: Der Irrgarten

Journey – eine Reise ins Ungewisse

Auf der Suche nach dem Magier Astrix, der für uns die einzige Rettung darstellt, bin ich, Tag, aus unserer Gruppe auserwählt worden, unsere Erlebnisse schriftlich festzuhalten.

Beginnen möchte ich im letzten Dorf vor dem Ungewissen, da wir hier vor unseren ersten Entscheidungen standen.

Wir besuchten zunächst ein kleines Geschäft. Hier wirkte eine Karte besonders anziehend auf uns. Der Händler war zwar der Meinung, daß sie für uns vollkommen ohne Wert sei, aber wir nahmen sie trotzdem mit. Als nächstes begaben wir uns in die kleine Taverne, in der Esher sich einmal genauer umsah, während wir anderen uns den Getränken zuwandten. Beim Verlassen fragte ein Dorfbewohner, ob er uns nicht begleiten könne, da er sich in der Gegend gut auskennen würde.

Bergon, unser Anführer, entschied, daß ein Scout von Nutzen sein könnte, und so kam es, daß Minar sich uns anschloß.

Wir verließen also das Dorf und kamen bald an eine Weggabelung. Minar kundschaftete den Weg aus und brachte von einem der beiden Wege einen Stoffetzen mit, der auf Räuber schließen ließ. Trotzdem beschlossen wir, diesen Weg zu nehmen. Vorsichtig schlugen wir am Abend unser Lager auf und waren nicht überrascht, als wir überfallen wurden. Minar flankte sofort zur Seite weg und umrundete auf diese Weise die Banditen, die wir nach einem kurzen Kampf ohne Verluste besiegten.

Wir folgten den Banditen und kamen an ein Hünengrab, in dem die Leichen der letzten verschollenen Expedition lagen. Nur ihr Anführer fehlte. Beim Untersuchen der Leichen fanden wir geringe Mengen magischer Essenzen, die Praxis sehr willkommen waren.

Unsere kleine Gruppe blieb immer auf dem Hauptweg, wir ließen Minar aber jede Abzweigung auskundschaften. Als er einmal länger als eine Stunde nicht zurückkam, machten wir uns auf die Suche nach ihm.

Minar fanden wir nicht, dafür entdeckte ich etwas Seltsames, das Praxis als Anfang zu der als Legende geltenden Zwergenstraße identifizierte. Er öffnete es und wir betraten zögernd die dunklen Höhlen.

Schon nach wenigen Metern hörten wir seltsame Geräusche, die sich wie Schritte anhörten. Also harrten wir der Dinge, die da kommen sollten. Bald erschien eine kleine Gruppe von Zwergen, der wir friedlich begegneten.

Der Anführer stellte sich als Agrith vor, der König der Zwerge. Als er sich nach unserem Anliegen erkundigte, erzählten wir ihm selbstverständlich die Wahrheit. Agrith zeigte sich sehr kooperativ und stellte uns seinen Sohn Hurth als Führer zur Verfügung.

Mit Hurth in der Führung machten wir uns also auf den Weg. Schon bald kamen wir an eine Treppe, und von Neugierde besessen, erklommen wir die Stufen. Nach einer sehr langen Zeit teilte sich die Treppe in zwei unterschiedliche Richtungen.

Neugierig geworden, folgten wir der Treppe, die laut Hurth im Sunset-Turm enden sollte. Nach einem sehr langen Weg fehlte auf einmal ein Stück der Treppe. Während wir uns noch Gedanken darüber machten, wie wir diesen Spalt überqueren könnten, ließ Praxis Esher mit dem Elevation-Spruch in die Luft steigen und blies ihn anschließend mit Wind über den Abgrund. Wir warfen Esher das Seil zu und konnten auf diese Weise die Schlucht überqueren. Damit wir auch wieder zurück konnten, wartete Bergon an der Treppe. Oben im Turm habe ich ein kaputtes Fernglas gefunden, welches ich natürlich an mich nahm.

Nach einiger Zeit kehrten wir um und drangen tiefer in die



Bild 3: Die Kanalisation von Xak Tsaroth

Bergon, der versuchte, Fische für uns zu fangen. Da die Suche nach ihm erfolglos blieb, gingen wir weiter, bis wir auf eine kleine Höhle stießen. Praxis sprach einen Glow-Spruch auf seinen Zauberstab, und so hatten wir eine Fackel. Im Innern der Höhle stießen wir auf einen See, in den ich, in der Hoffnung Bergon zu finden, sprang. Tauchend durchquerte ich eine kleine Höhle und schwamm in einer anderen wieder an die Oberfläche. Kaum war ich aufgetaucht, sah ich Bergon gefesselt und verletzt am Boden liegen und versteckte mich. Nach kurzer Zeit erschien eine Nymphe und sah nach Bergon. Als sie weg war, befreite ich ihn, ließ ihn aber, da er sehr geschwächt war, liegen. Ich erforschte die größere der beiden Höhlen und betrat zunächst die linke Höhle, in der ich eine Fackel aufnehmen konnte. Mit der Fackel ging ich in den Raum rechts, der sich als Schatzkammer herausstellte. Ich sah ein blaues Amulett, welches ich an mich nahm. Da ich, wie ich schnell feststellte, mit dem Amulett die Schatzkammer nicht verlassen konnte, besuchte ich den Nebenraum, hob die Platte ab und warf das Amulett in den Teich. Danach suchte ich den Raum mit der Fackel auf und sprang in den Teich. Die Fackel brannte auch unter Wasser, und ich sah seltsamerweise das Amulett hier liegen. Ich nahm es an mich, schnappte Bergon und verließ mit Bergon und dem Amulett die Höhle.

Wir setzten unsere Reise fort, und es dauerte nicht lange, da passierte der nächste Zwischenfall: Minar wurde von einer Schlange gebissen, und seine einzige Rettung war ein Kraut, das es jetzt zu suchen galt.

Also teilten wir uns auf. Bergon kehrte bald wieder um, als er merkte, daß es auf seinem Weg nicht weiterging. Praxis fand einen hohlen Baumstumpf, den er mittels eines Blitz-Zaubers öffnete.

Die entstandene Höhle untersuchte er natürlich sofort und stellte fest, daß die Gänge unterirdisch in die Richtung führten, aus der er gekommen war, und in der auch der verletzte Minar lag. Praxis bewegte sich in der Höhle in Richtung auf die Unfallstelle zu und fand neben einem weiteren Ausgang auch noch das gesuchte Kraut.

Zwergenstraße ein. Wir stießen auf ein Tor, auf dem 'Gefängnis' stand und Hurth öffnete es für uns. Nachdem wir es durchquert hatten, ließ es sich von innen aber nicht mehr öffnen. Also suchten wir nach einem anderen Ausgang. Da ich schon nach kurzer Zeit das Gefühl bekam, ständig im Kreis zu laufen, legte ich einen Gegenstand auf den Boden. Und richtig! Mein Verdacht bestätigte sich sehr schnell.

Praxis fand dann aber mit Hilfe eines Zauberspruches einen Ausgang nach oben, den wir wieder einmal mittels Elevation erreichten.

Nicht weit entfernt gabelte sich der Weg. Hurth fand heraus, daß beide Wege zu einer Gruppe von Orcs führten, der rechte Weg aber mehr Deckung bot. Wir schlichen uns also an die Orcs heran und stellten fest, daß sie Minar gefangen hatten. Es stand außer Frage, daß wir ihn retten mußten. Die Orcs waren aber leider in der Überzahl, so daß wir uns eine besondere Taktik einfallen lassen mußten. Hurth schlich sich also davon und griff von hinten an. Dadurch hatte Praxis genügend Zeit, einen Schlammzauber auf den Boden zu werfen, so daß sich die Orcs nur sehr schwer bewegen konnten. So schafften wir es, die Orcs zu besiegen; zwei von ihnen konnten allerdings fliehen. Hurth nahm sofort die Verfolgung auf. Wir befreiten zunächst Minar und folgten dann Hurth. Dieser lag sehr schwer verletzt am Boden, die Orcs hatten ihn in einen Hinterhalt gelockt. Bald kam sein Vater und ließ Hurth, der dem Tode nahe war, davontragen. Er gab uns ein Amulett für Astrix, und wir machten uns auf den Weg ans Tageslicht. Draußen schlugen wir unser Lager auf und gedachten noch lang des heldenhaften Todes Hurths.

Am nächsten Morgen machten wir uns wieder auf den Weg. Minar hatte beim Erforschen der Gegend den Rauch eines Kaminfeuers gesehen.

Wir folgten dem Rauch und trafen bald auf eine sehr baufällige Hütte, deren einziger Bewohner sich als Anführer der letzten Expedition erwies und unter dem Druck der Verantwortung geistig zusammengebrochen war. Da wir auch ihn nicht belogen, warf er uns beim Verlassen der Hütte ein Säckchen mit magischen Essenzen nach.

Als nächstes kamen wir an ein kleines Flüßchen, auf dessen Boden Esher Gold zu sehen glaubte. Wir suchten also nach Gold. Aber da sich ein Unwetter ankündigte, ließ Praxis einen Elevation-Spruch auf mich los, der mich und mein Gepäck vor einer herannahenden Flutwelle beschützte.

Praxis stellte fest, daß Eshers Gold Wasseressenz war und verstaute sie in seinem Beutel. Der nächste, der aus unserer Gruppe verschwand, war

Ich selbst begegnete einer Waldelfin, der ich mich vorsichtig näherte. Ich sprach sie an, aber sie schien nur elfisch zu verstehen. Die einzigen elfischen Sätze, die ich beherrschte, kannte ich aus Hurths Geschichte über die Elfen. Ich reimte mir die Worte "Agrith bran" zusammen, was soviel bedeutet wie "Ich bin ein Freund Agriths" – und sie verstand mich! Sie sprach mich in meiner Sprache an und begrüßte mich. Dann folgte sie mir zu Minar, den sie auch gleich heilte. Wir beschlossen, das Elfenlager aufzusuchen, wollten jedoch vorher Praxis' Höhle erforschen. Gesagt, getan. Unten in der Höhle fand Praxis noch etwas Feueressenz, aber dann erlosch sein Zauberstab langsam, so daß wir zwar im Dunkeln standen, aber einen Ausgang über uns entdeckten.

An unserer ersten Begegnungstätte trafen wir die Elfin wieder und begleiteten sie. Kaum im Lager angekommen, erreichte uns die Nachricht, daß der Wald vor uns brennt.

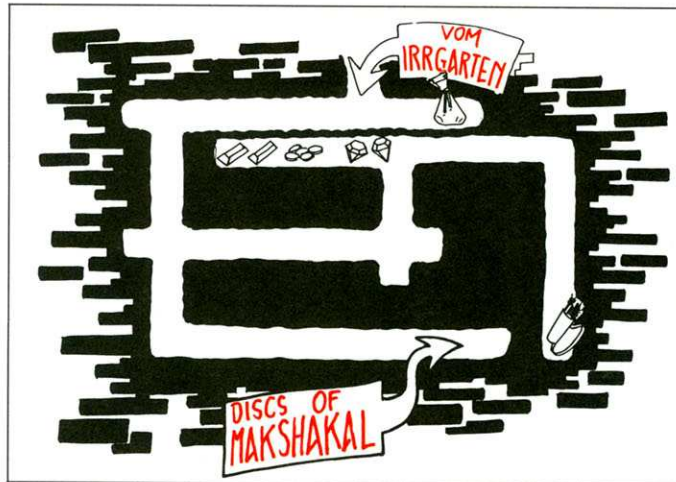


Bild 4: Die letzten Gänge, hier finden Sie die Disks von Makshakal

Wir beschlossen natürlich zu helfen.
An der Brandstelle angekommen, sprach Praxis einen Regenzauber mit all seiner Wasseressenz. Die einsetzende Regenflut löschte den Brand. Nach einem Fest gaben uns die Waldelfen aus Dank ein grünes Amulett. Bald brachen wir erneut auf und kamen an einen Fluß. Wir folgten dem Flußlauf stromaufwärts, soweit es nur ging. Hier bauten wir ein Floß, mit dem wir trotz der Einwände Eshers den Fluß überquerten und gerade noch heil am anderen Ufer ankamen. Beinahe wären wir nämlich einen Wasserfall hinuntergestürzt.

Jetzt standen wir vor einem Berg, an dem sich unzählig viele Wege nach oben schlängelten. Bei diesem Anblick kam uns der Gedanke an die im Laden erstandene Karte. Zunächst fanden wir uns mit der Karte überhaupt nicht zurecht, doch irgendwann kam Praxis die Idee, es doch einmal mit einem Glow-Spruch zu versuchen. Es funktionierte prächtig: Der Spruch ließ die ursprünglichen Runen aufleuchten, die uns den Weg zu Astrix' Turm zeigten.

Astrix zeigte sich erfreut über unseren Besuch und besonders darüber, daß wir die drei Amulette mitgebracht hatten. Er selbst besaß nämlich den vierten roten Stein der Magier. Nur mit den vier Steinen kann man die zwei finden, die dann schließlich zu dem einen Stein führen, mit dem das Böse, das die Welt bedroht, vernichtet werden kann. Minar verabschiedete sich von uns. Er wollte heim und sei-

TOOLBOX EDITION 68000

Wer seinen Computer oder Anwendungen programmieren will, findet hier Tips, Anregungen und vor allem Anwendungen erster Güte.

Das Projekt Fraktal wartet mit einem Weltrekord auf:
Das "Apfelmännchen" wird in 7 Sekunden auf den Bildschirm eines Amiga 2000 gezeichnet!! Da waren Hexenmeister am Werk.

Viele Reviews informieren über neue Programmierer-Software und verhelfen dem Leser zum Überblick über interessante Neuerungen.

Natürlich gibt es die Programme des Heftes auch in der DATABOX. DATABOX ist der Software-Service, der bei DMV Standard ist. Wem das Tippen zu fehlerträchtig ist, der bestellt die DATABOX zum Heft.

Aus dem Inhalt:
PROJEKT FRAKTAL – Portierung in M2Amiga
WELTREKORD – Apfelmännchen in 7 Sekunden
ACU – Mausgesteuerter Compiler
C-Compiler-Oberfläche
Universeller Multitasking-FileRequester in M2Amiga
Undercover Viruskiller mit eigenem Task
Do-Request – Trickreicher AutoRequester
AmigaDOS – Fenster mit Zwittereigenschaften
M2Amiga-Inline-Code per Programm

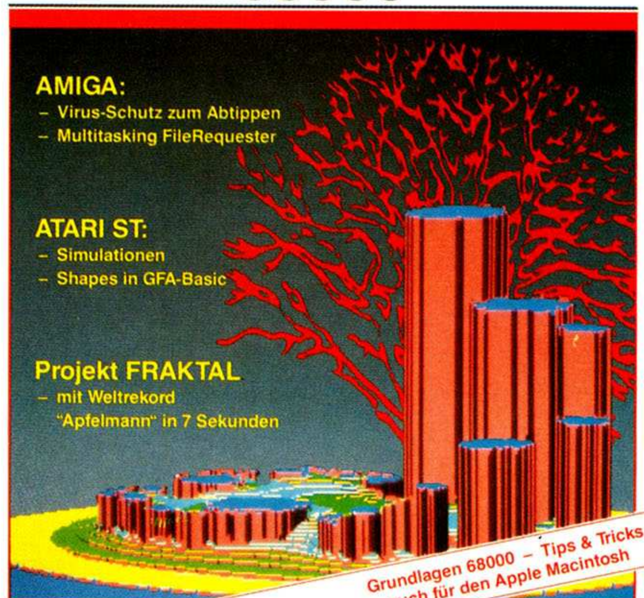
DATABOX Amiga Best.-Nr. 3502 **29,– DM***

TOOLBOX EDITION 68000 Best.-Nr 3501
Sonderpreis für AMIGA-DOS-Leser **9,– DM***

* Unabhängig von der Anzahl der bestellten Produkte berechnen wir für das Inland 4,– DM bzw. für das Ausland 6,– DM Porto und Verpackung.

Bitte benutzen Sie die Bestellkarte

DMV-Verlag · Postfach 250 · 3440 Eschwege



nen Leuten alle Erlebnisse und Abenteuer berichten. Wir allerdings machten uns auf die Suche nach den zwei Steinen...

Sobald es von Journey und der Suche nach den letzten verbliebenen Steinen Neues zu berichten gibt, werden wir Sie auf den Spieletips-Seiten davon unterrichten.

Ein richtiger Cheatmode mit unendlichen Leben oder Unverwundbarkeit ist es leider nicht.

Dirk und der Drache Cheatmode zu Dragons Lair

Aber, wird dieser Trick angewandt, dann spielt sich das Spiel wie ein Zeichentrickfilm ganz von alleine durch. Sie brauchen nur folgende Tastenkombination gleichzeitig zu drücken: <ESC>, <R>, <L>, <N>, <O> und (-). Wenn alles geklappt hat, blitzt der Bildschirm kurz auf. Das Spiel befindet sich im Cheat-Mode, und Sie brauchen sich nur noch zurückzulehnen und zuzuschauen.

Manhunter New York

Sie stecken in der Kanalisation von New York? Der Gestank geht Ihnen auf die Nerven, und Sie würden gerne so schnell wie möglich alle Codekarten finden und entkommen? Ihr Wunsch ist gar nicht so schwer zu erfüllen. Sehen Sie sich doch einmal unsere Karte des Kanalisationssystems an. Hier ist alles, was Sie suchen, eingezeichnet.

Ooze – Der Dauergrusler aufgeknackt

Die Erbschaft des Cheez Burger ist geisterverseucht und höchst problematisch. Um dem Spuk ein Ende zu bereiten, haben wir hier einige Hilfen zusammengestellt. Wenn man sich auf der Veranda in den Schaukelstuhl setzt, steigt der Mut-Wert. Also nichts wie rein in den Stuhl und ein wenig herumgeschaukelt! Vergessen Sie aber nicht die herausgebrochene Strebe des Verandageländers. Sobald man das Haus betritt, wird man angegriffen, doch keine Angst, bleiben Sie einfach stehen. Der Erfolg: eine

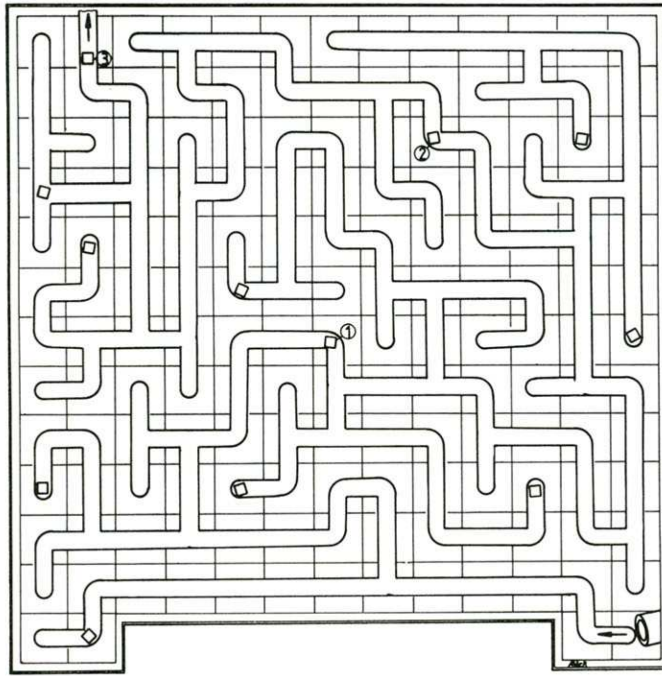


Bild 5: Man Hunter New York – Genauer Plan des Kanalsystems

weitere Erhöhung des Mutes um zwei Punkte.

Die mittlere Tür in der Passage sollte man erst öffnen, wenn die Türklinke heruntergedrückt wurde. Dadurch wird ein Geheimmechanismus ausgelöst (Türklinke drücken, Tür öffnen und nach Norden gehen). Binden Sie nun das Seil los und nehmen Sie es in Ihr Inventar auf. In der Kommode hockt Ludicrus Geist (Achtung, der Mut-Wert sinkt!).

Um den Eispickel aus der Truhe in der Passage zu holen, brauchen Sie kein Licht (Truhe öffnen, Pickel herausnehmen). Wenn sich der Computer dagegen sträubt, sollte man diese Befehle einfach mehrmals eingeben. Andererseits können Sie sich auch der Glühbirne im Musikzimmer bemächtigen und diese dann in der Passage austauschen.

Die Rüstung in der Halle attackiert den Spieler, wenn er sie mitnehmen will. Folge: Die Gesundheit sinkt um einen Punkt.

Im Speisesaal besorgt man sich eine Axt und tötet Kunibert. Dadurch erhöhen sich der Mut- und der Charisma-Wert. Man kann jedoch auch abwarten, bis ein Geist erscheint und dem armen Kunibert einen Eimer Wasser über die Rüstung schüttet. Diese rostet augenblicklich und hält Kunibert gefangen. Um ihn wieder zu befreien, benötigen Sie etwas Öl, das Sie sich in der Küche holen können. Im Raum befindliche Ratten werden mit Lebensmitteln abgelenkt.

Im Schlafzimmer sollte man den Frisiertisch beiseite schieben und die Wand mit dem Pickel einschlagen. Durch diese Aktion wird ein weiblicher Geist befreit. Folge: Mut-, Charisma- und Gesundheitswerte steigen. Wenn man den Geist anschaut, steigt der Charisma-Wert wiederum.

Das Seil bindet man an die Axt und wirft diese über das Loch in der Treppe nach oben. Nun kann man an dem Seil nach oben klettern (man benötigt hohen Mut-Wert).

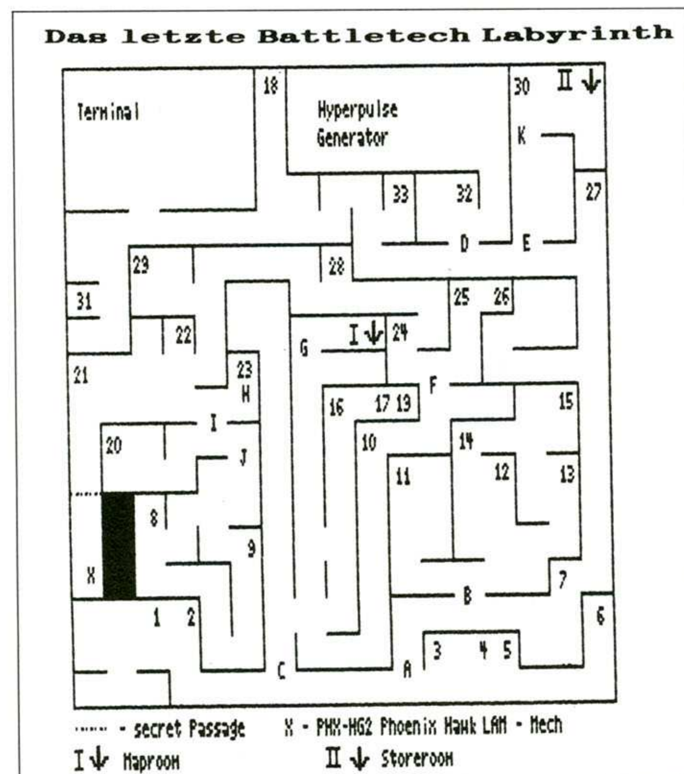


Bild 6: Battlefinale – Anhand dieser Karte des letzten geheimen Labyrinths gibt es auch hier keine Probleme mehr

The Kristal – Kampfhilfe für schlechte Schwertkämpfer

Haben Sie nicht auch öfters das Problem, daß die Gegner auf den fremden Planeten Ihnen stets den Garaus machen, da jene wesentlich stärker und schneller sind?

Ganz klar, eine bessere Waffe muß her, doch woher nehmen, wenn nicht stehlen? Ganz einfach. Geben Sie dem armen Bettler Bendoon auf Meltoca zweimal hintereinander ein Geldstück. Er gibt Ihnen als Gegenleistung einen Schwertknauf, der den ersten Teil einer Superwaffe darstellt. Auf dem Planeten Darn finden Sie dann die Schneide dazu, die Sie auf den Knauf stecken müssen. Nun ist es wesentlich einfacher, den Gegner zur Strecke zu bringen.

(Robert Marz/hs)



Bei diesem Spiel können Sie die gefährlichste und aufregendste Achterbahnfahrt Ihres Lebens erleben

Roller Coaster Rumbler

Gehören Sie auch zu denjenigen, die auf jedem Rummelplatz zu finden sind, auf dem es eine Achterbahn gibt? Macht es Ihnen Spaß, gebeutelt und durchgeschüttelt zu werden? Sind Sie schon soweit, daß Sie während der Fahrt etwas anderes, Actionreicheres tun möchten?

Rasante Achterbahnfahrten verbunden mit Vektorgrafik und viel Ballerei ist die Grundlage für eine neue Spielidee. Allein oder zu zweit schießt man dabei gleichzeitig auf Ziele oder Hindernisse, die immer wieder auf den Schienen der Achterbahn liegen. Der Wagen, in dem man fährt, kann beschleunigt oder gebremst werden. Allerdings ist Tempo angesagt, da Sie gegen die Zeit fahren. Erst wenn Sie eine bestimmte Bestzeit unterboten haben, geht es im zweiten Level weiter. Geht es am Anfang noch recht gemütlich zu, da nur Übungsziele auf der Strecke liegen, kommen sehr bald die ersten schwingenden Kugeln auf Sie zu. Diese sind an Toren aufgehängt und schwingen über der Strecke hin und her. Dann liegen die ersten Minen auf der Strecke, die fleißig von kleinen Flugzeugen abgeworfen werden. Diese gilt es abzuschießen, da sich sonst der Zustand des Wagens durch das Berühren der Minen stark verschlechtert. Der Spieler kann die Blickrichtung verändern und wahlweise nach vorne oder hinten aus dem Wagen blicken und natürlich auch in diese Richtungen schießen. Das Fadenkreuz des Gewehrs wird mit der Maus gesteuert, ein eventuell teilnehmender zweiter Spieler muß zum Joystick greifen. Action gibt es in Roller Coaster wahrlich genug. Durch den schnell steigenden Schwierigkeitsgrad bleibt das Spiel lange eine Herausforderung.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Roller Coaster Rumbler
Hersteller: Tynesoft
Quelle: Fachhandel
Preis: 84,95 DM

Positiv:
- 3D-Vektor-Grafik
- Zweispielermodus
- viel Action

Negativ:
- spartanische Bedienungsanleitung





Terrys großes Abenteuer führt Sie in einen verwilderten Garten, in dem schon jede Menge Ungeziefer auf Sie wartet

Terry's Big Adventure

Während die meisten Leute auf ihrem Weg nach Hause bestenfalls Abenteuer mit einer Ampel, die ewig rot ist, erleben können, muß Terry einen großen, verwilderten Garten durchqueren, und hier lauern mehr Gefahren, als Terry lieb ist.

Der bewußte Garten ähnelt einem dichten Dschungel und statt Tiger und Löwen durchstreifen gefährliche Igel, gefräßige Raupen und greuliche, blutrünstige Eulen das grüne Dickicht. Aber der kleine Terry ist nicht wehrlos, und was dem Großwildjäger die Elefantenbüchse, ist für Terry sein Jojo. Dieses schwingt er um sich und kann damit das Ungeziefer verjagen. Abgesehen von den gefährlichen Biestern muß sich Terry auch vor einigen anderen Objekten fernhalten. Grüne Giftpilze und brackige Wasserlöcher bedrohen sein Leben und können nur durch geschicktes Springen vermieden werden. Aber nicht nur Gefahren lauern, es verbirgt sich auch Angenehmes im Garten. Rote Pilze, die beim Einsammeln Terrys Punktekonto verbessern, oder wertvolle Steine, die Terry allerdings erst einmal aus dem Felsen schlagen muß.

Der Garten ist in zwölf Teilstücke unterteilt, die Terry nacheinander durchwandern muß. Zwischen jedem Gartenstück liegt eine Bonusrunde, in der Terry Pilze sammeln kann, ohne dauernd von Gefahren gestört zu werden.

Terry's Big Adventure ist ein Hüpf- und Sammelspiel traditioneller Machart. Neben Offensichtlichem haben die Programmierer noch einige Extras, wie z.B. Geheimlevel, eingebaut. Die Grafik des Spiels ist mager und Besseres wäre technisch sicher machbar gewesen. Aber dafür ist der Sound recht gelungen und über mangelnden Spielwitz und Motivation kann nicht geklagt werden.

(hs)

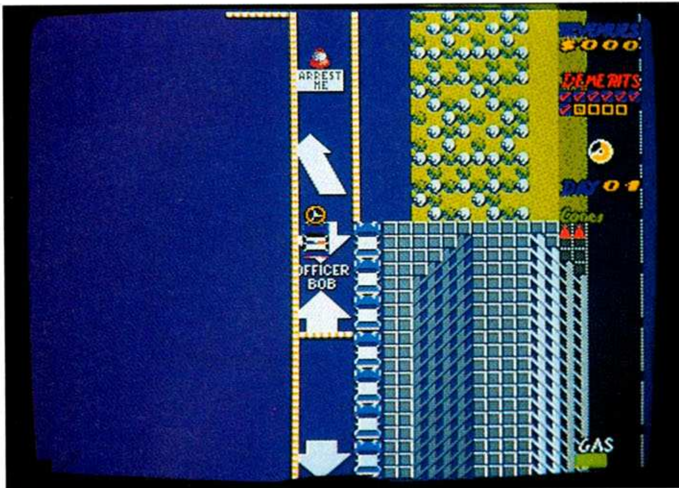
AMIGA DOS Blitzlicht

Name: Terry's Big Adventure
Hersteller: Shades
Quelle: Fachhandel
Preis: 49,95 DM

Positiv:
- versteckte Extras
- gute Musik

Negativ:
- magere Grafik
- zu wenig Levels





Als Polizist durch die Stadt fahren und Strafzettel verteilen, das wär' doch mal was, oder?

APB

Großstadtpolizisten haben keinen leichten Job. Von der Verkehrssünde bis zur Umweltverschmutzung reicht das Spektrum der Delikte, die so ein Cop verfolgen muß. Und als ob es nicht genügen würde, gibt es da auch noch einen cholerischen Chef, der jeden Fehler auf die Goldwaage legt.

Hat man erst einmal im Streifenwagen Platz genommen, ist es schon erstaunlich, wie viele Delikte es zu ahnden gilt. Um schädlichem Müßiggang vorzubeugen, hat Ihr Chef Ihnen eine Quote vorgegeben. Im Klartext, Sie müssen für mindestens 2000$ Ordnungswidrigkeiten ahnden. Gelingt dies nicht, bekommt Ihr Chef einen Anfall und zweifelt an Ihren Qualitäten. So weit wollen wir es aber nicht kommen lassen, fahren durch die Straßen und halten die Augen offen. Als guter Hilfscop in der Lehre fangen wir ganz gemächlich an. Der erste Auftrag besteht darin, einige Umweltverschmutzer zu fassen. Vor dem Revier wartet der Polizeiwagen, mit dem Sie durch die in der Vogelperspektive dargestellte Stadt chauffieren. Das Einfädeln in den fließenden Verkehr geht noch halbwegs ohne Probleme, aber einem Sünder während der Fahrt ein Ticket zu verpassen, bedarf einiger Übung. Für jeden Fehler, den Sie zu verantworten haben, erhalten Sie eine Disziplinarstrafe. Natürlich müssen Sie während Ihres Streifendienstes auch nach gesuchten Verbrechern Ausschau halten. Gelingt es Ihnen, einen dieser Typen zu fassen, winkt ein Extrabonus. Nett, anders kann man's nicht umschreiben. APB spielt sich gut und abwechslungsreich, ist technisch in Ordnung, probat gestaltet und alles in allem ein recht friedliches Spiel. Ich sehe keinen Grund, warum ich Ihnen dieses Spiel nicht empfehlen sollte.

(hs)

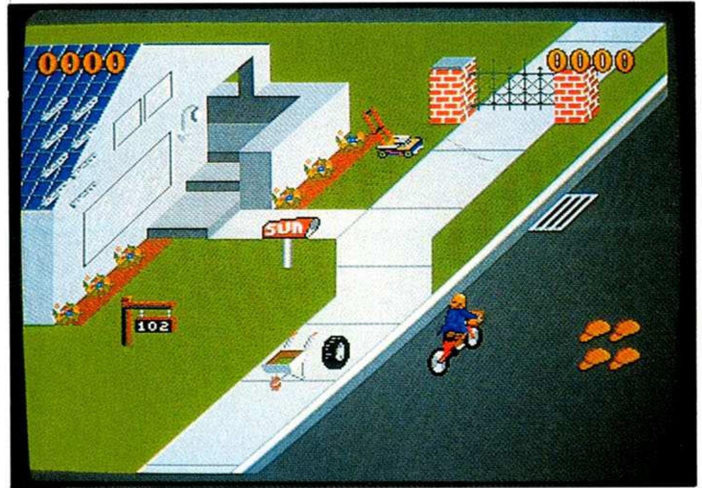
AMIGA DOS Blitzlicht

Name: APB (All Points Bulletin)
Hersteller: Domark
Quelle: Fachhandel
Preis: 64,95 DM

Positiv:
- interessantes Spielkonzept
- Sound und Grafik gut

Negativ:
- schlecht übersetzte Anleitung





Wie viele Kunden werden Sie nach der ersten Tour noch haben?

Paper Boy

Eine Tageszeitung, das ist für viele Menschen ein unverzichtbarer Bestandteil des Frühstückstisches. Doch von der Druckerei bis zum Frühstückstisch ist es ein weiter Weg, der auch irgendwie bewältigt werden muß.

Da kommt der Zeitungsträger ins Spiel, so wie er hierzulande genannt wird. Im Land der unbegrenzten Möglichkeiten nennt man die Burschen Paper Boys, und schlußendlich spielen wir ja einen US-Zeitungsjungen.

Mit einem Fahrrad und einem Stapel Zeitungen geht es los. Es gilt die tägliche Tour so schnell und mit so wenigen Fehlern wie möglich zu absolvieren.

Der flotten Zustellung willen macht sich der Paperboy erst gar nicht die Mühe vom Rad zu steigen, sondern wirft seine Zeitungen aus voller Fahrt in die Briefkästen. Allerdings will dieses Kunststück gelernt sein, da jeder Fehlwurf Kunden kostet, oder lesen Sie gerne Zeitungen, die in einem Hundehaufen gelandet sind? Aber das ist noch längst nicht das einzige Handicap, das dem Paper Boy das Leben schwer macht. Da gibt es noch rabiate Pudel, wildgewordene Rasenmäher, verschlossene Gartentüren und Verkehrsrowdies, die wie angestochen aus Seitenstraßen rasen. Eine Kollision mit einem dieser Hindernisse wirft den Paper Boy aus dem Sattel und kostet wertvolle Zeit. Von Ihren Leistungen hängt es ab, wie viele Anwohner aus Ihrem Bezirk auch in Zukunft ihre Zeitung von Ihnen geliefert bekommen möchten. Also, je besser Sie die Tour absolvieren, desto größer der Erfolg.

Wer auf der Suche nach einem friedlichen Geschicklichkeitspiel ist, sollte einmal einen Blick auf Paper Boy werfen. Grafik und Sound sind zwar eher mittelmäßig, aber dafür macht das Spiel viel Spaß.

(hs)

AMIGA DOS Blitzlicht

Name: Paper Boy
Hersteller: Elite
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- sauberes Scrolling
- viel Witz im Detail

Negativ:
- Grafik mittelmäßig





An die einzelnen Steine ist nicht immer ganz einfach heranzukommen. Oft muß man ganz schön knobeln



Scheinbar lernen die Aliens es nie, zum x-ten Male ist die Erde Invasionsziel

Minos

Labyrinthspiele haben die meisten Menschen seit jeher begeistert und sind seit Pacman auch auf Homecomputern allgemein beliebt. Sie laufen immer nach einem ähnlichen Schema ab...

Es gilt eine Aufgabe zu erfüllen: den Ausgang zu finden und in das nächste Level zu schlüpfen. Bei Minos verhält es sich ähnlich. Es können allerdings bis zu zehn Spieler an einem Spiel teilnehmen.

Bei Minos hat man, bevor man einen der fünfzig Levels als abgeschlossen betrachten kann, eine ganze Menge zu tun. Es gilt nämlich, eine bestimmte Anzahl von Steinen einzusammeln. Daß die Programmierer dem Spieler das Leben so schwer wie möglich gemacht haben, dürfte jedem einleuchten. Die einzelnen Labyrinthe setzen sich aus verschiedenen Elementen zusammen: Wände, Steine, Fallen, Türen und Schlüssel, Einbahnstraßen, Säulen, die nach dem Überqueren einstürzen und zu Mauern werden, Teleportfelder, ein Geheimnisfeld und der Ausgang. Läuft man über eine Falle, so verliert man alle Schlüssel und muß einen Stein mehr aufsammeln.

Stellt man fest, daß man nicht weiterkommt, so reicht ein Druck auf den Feuerknopf des Joysticks, um den Level von vorne zu beginnen. Innerhalb der ersten fünfzehn Levels gibt der Computer nach dem dritten mißlungenen Versuch eine kleine Hilfestellung. Häufig wechselt das Aussehen der Labyrinthe und der Spielfigur. Damit man sich aber trotzdem zurechtfindet, ist am unteren Bildschirmrand eine Legende angebracht, auf der die einzelnen Symbole erläutert werden.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Minos
Hersteller: Turtle Byte
Quelle: Fachhandel
Preis: 69,- DM

Positiv:
- durch 50 Levels hohe Motivation
- gute, nicht störende Hintergrundmusik
- Abwechslung durch sich verändernde Grafik
- regelbare Geschwindigkeit der Spielfigur
- mitgelieferter Editor

Negativ:
- auch in schnellster Stufe noch recht langsam



Slayer

Wer Action sagt, der meint zumeist Ballerspiele. Dieses Genre, das sich auf dem Amiga großer Beliebtheit erfreut, hat mit Slayer einen neuen Vertreter bekommen.

Slayer ist ein horizontal scrollendes Ballerspiel, bei dem es mal wieder darum geht, garstige Aliens zu vertreiben. Diese haben sich in einigen Weltraumzonen festgesetzt und dort reichlich Verteidigungs-anlagen installiert. Es gilt also, die Zonen möglichst wohlbehalten zu durchfliegen. Neben Geschütztürmen, die am oberen bzw. unterem Bildschirmrand postiert sind, gibt es natürlich auch eine ganze Menge fliegender Bösewichter, die auf diesem Wege versuchen, ihr Raumschiff zu zerstören.

Zu Anfang des Spiels ist der Raumjäger mit einem Laser eher mager ausgestattet. Beim tieferen Eindringen in die Alienzonen tauchen immer wieder Container auf, mittels derer die Feuerkraft des Raumschiffs erhöht werden kann. Zu dem normalen Laser können auf diese Weise Diagonalschüsse und ein Satellit, der das Schiff umkreist, gewonnen werden.

Insbesondere der Satellit hat sich in der Praxis als sehr nützlich erwiesen, da er einerseits selbst schießt und andererseits gegnerische Schüsse abfängt.

Hat der Spieler eine Zone bis zu deren Ende durchflogen, wird er dort vom obligatorischen Obermonster erwartet, das den Zugang zum nächsten Level bewacht. Slayer braucht sich nicht zu verstecken. Die Grafiken sind ausgezeichnet, der Sound gut, wenn auch die Sound FX etwas einfallslos gerieten. Die Extrawaffen erweisen sich dem Spiel und dessen Schwierigkeitsgrad angemessen.

Slayer ist ein preiswertes, gut gemachtes, und sehr spielbares Programm geworden, das beweist, daß gute Ballerspiele nicht unbedingt auch teuer sein müssen.

(hs)

AMIGA DOS Blitzlicht

Name: Slayer
Hersteller: Hewson
Quelle: Fachhandel
Preis: 64,95 DM

Positiv:
- gute Grafik
- viel Spielspaß
- preiswert

Negativ:
- einfallslose Geräuscheffekte





Pharao sein, das ist nicht schwer, Pharaoh werden, dagegen sehr!

Day of the Pharao

Vor 4000 Jahren lebten die alten Ägypter am Ober- und Unterlauf des Nils. Die Pyramiden, Baudenkmäler von gewaltigen Abmessungen, künden noch heute von der hochstehenden Kultur.

Vor dem Hintergrund dieser schillernden Kultur ist 'Der Tag des Pharaos' angesiedelt. Der Spieler übernimmt die Rolle eines jungen Prinzen. Nach dem Tode des herrschenden Pharao werden alle Thronfolger ermordet. Nur eben jener Prinz entgeht dem Zugriff der Meuchelmörder. Er wächst in einem sicheren Versteck auf, und als er alt genug geworden ist, schlägt er die Laufbahn eines Händlers ein und versucht, auf diese Weise das Geld zusammenzutragen, das benötigt wird, um wieder den Thron zu besteigen.

Mit einer mickrigen Nilbarke beginnt der Spieler seine Laufbahn. Es gilt, zwischen den einzelnen Nilmetropolen Handel zu treiben, so wenig wie möglich an die phönizischen Piraten zu verlieren, und so viel Profit wie möglich zu machen: neue Schiffe zum Beispiel oder Weihgeschenke für die Götter, damit diese Ihren Plänen wohlgesonnen bleiben. Day of the Pharao ist eine etwas weiter gefaßte Version einer Handelssimulation mit einigen Strategieelementen. In die Handlung wurden noch Actionsequenzen eingeflochten. So müssen Piratenangriffe mit dem Joystick in der Hand abgewehrt werden, je schlechter man dabei abschneidet, desto leerer ist der Laderaum.

Day of the Pharao kann zwar einige nette Ansätze verbuchen, bleibt aber trotz allem in der Mittelmäßigkeit stecken. Die Grafik ist entweder digitalisiert oder mager, der Sound ist haarsträubend und die Ladezeiten horrend. Wenn Sie sich Pharao nicht kaufen, haben Sie nur wenig verpaßt und einiges an Geld gespart.

(hs)

AMIGA DOS Blitzlicht

Name: Day of the Pharao

Hersteller: Rainbow Arts

Quelle: Fachhandel

Preis: 89,– DM

Positiv:
- interessanter Spielverlauf

Negativ:
- magere Grafik
- bescheidener Sound
- lange Ladezeiten





Erst auf die Nase, dann auf den Kiefer, Thaiboxen macht Spaß und tut weh

Bangkok Knights

Die Karatewelle ist noch lange nicht vorbei. Kung Fu und Kickboxen haben nach wie vor Hochkonjunktur. Wer keine Lust hat, blaue Augen und ausgeschlagene Zähne zu riskieren, kann nun gefahrlos einen Kampf wagen, Bangkok Knights macht's möglich.

Es gilt, die Bangkok Meisterschaft im Kickboxen zu gewinnen. Acht verschiedene Gegner stehen zwischen Ihnen und dem Sieg. Jeder der Kämpfer hat seinen individuellen Kampfstil, der durchaus auch unfaire Mittel beinhalten kann. Ihr Bildschirmfighter kann sich mit acht verschiedenen Schlägen und Tritten zur Wehr setzen.

Die Kämpfe, die siegreich überstanden werden müssen, bestehen aus fünf harten Runden.

Wer dem Sieg näher ist, kann in einer Anzeige abgelesen werden, die sich unterhalb des eigentlichen Spielscreens befindet. Je kürzer der dort befindliche Balken ist, desto schlechter geht es Ihrem Kickboxer.

Gerade Karatespiele erlebten vor einem Jahr einen unerhörten Boom, und dutzende Spiele sehr ähnlicher Machart erschienen in einem kurzen Zeitraum. Nur Amiga-Anwender wurden von diesem Boom weitgehend verschont. Bangkok Knights war und ist eines der besseren Karatespiele, auch wenn die nun vorliegende, etwas spät erschienene Amiga-Version an einigen Stellen zu wünschen übrig läßt.

Da ist z.B. die Grafik, die zwar ganz nett gezeichnet ist, aber bei weitem nicht Amiga-Standard erreicht. Der Sound ist zwar hörenswert, geht aber spätestens nach einer halben Stunde so auf die Nerven, daß der Anwender für die Option "Sound off" dankbar sein wird.

Ansonsten ist Bangkok Knights ein passables Karatespiel, das allen Freunden der Martial Arts einiges an Spaß bereiten wird.

(hs)

AMIGA DOS Blitzlicht

Name: Bangkok Knights

Hersteller: System 3

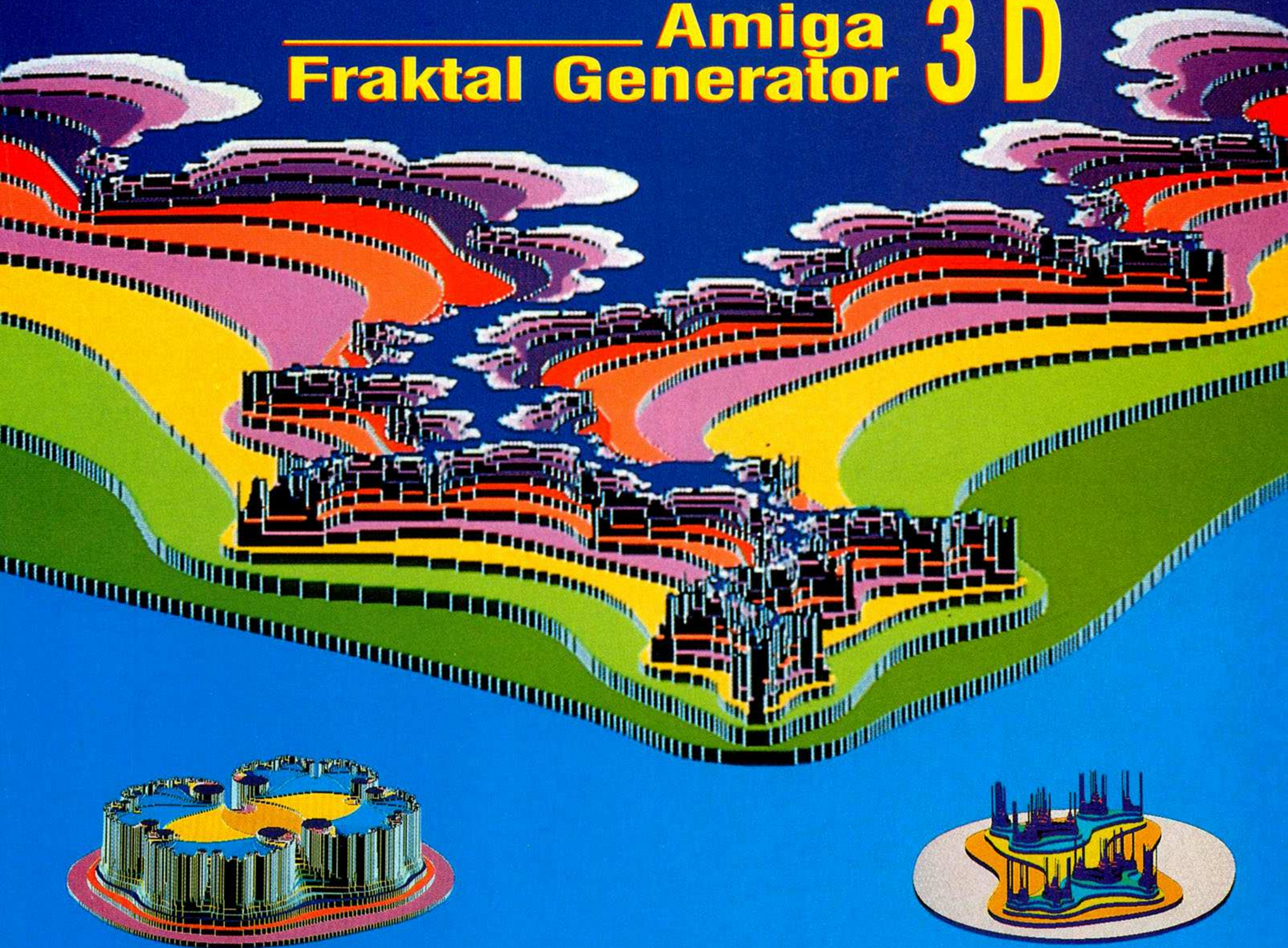
Quelle: Fachhandel

Preis: 84,95 DM

Positiv:
- verschiedene Gegner

Negativ:
- nervtötende Musik





Supergrafik im Sekundentakt

Vergessen Sie alles, was Sie bisher über Fraktalgrafik-Programme gehört haben – die unendliche Weite phantastischer Bilder erschließt sich nur über ein superschnelles Programm: **Fraktal Generator 3D**

High-Speed
Nur noch 7 Sekunden für das Urbild!

Super-Parallel-Projektion
Frei wählbarer horizontaler Blickwinkel mit 360 Grad:
Betrachten Sie das "Fraktalobjekt" von allen Seiten
Stufenloser vertikaler Blickwinkel:
Wahlweise Sicht von oben und unten, schräg oder in der Totalen

Speicherung im IFF-Standard
Einladen der Fraktal-Bilder in Mal- und Zeichenprogramme:
Verwendung als Hintergrund, Motiv oder Vorlage

Voller Bedienungskomfort
Auswahl komplett mit Pull-Down-Menüs
Wahlweise Maus- oder Tastensteuerung

Phantastische Farbmöglichkeiten
32 Farben im Low-Resolution-PAL-Modus
Eigenes Farbrequester mit stufenloser Schiebereglung

Mehrere separate Bildspeicher
Bis zu vier Bilder gleichzeitig abrufbar
Separate Farbzuordnung und Animationsmöglichkeit

Farb-Animationen
Phantastische Effekte durch Amiga-Color-Cycling

Amiga Fraktal Generator 3 D

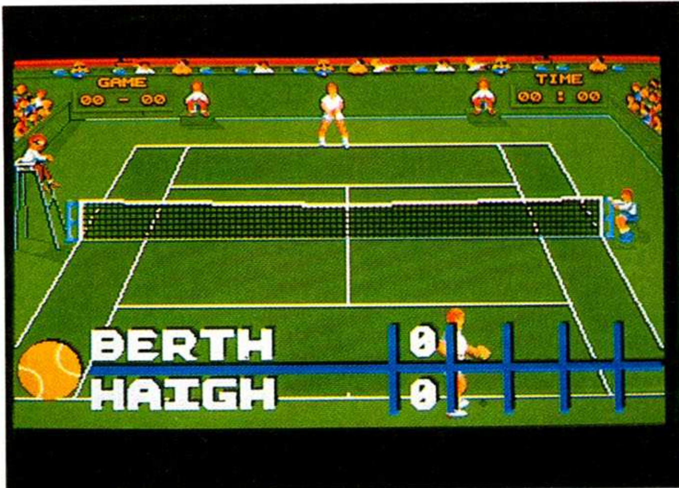
3 1/2"-Disk. Best.-Nr. 2901

69,- DM (unverbindliche Preisempfehlung)
Wenn Sie über den DMV-Bestellservice bestellen, gilt folgendes:

Inland:		**Ausland:**	
Einzelpreis	69,– DM	Einzelpreis	69,– DM
zzgl. Versandkosten	4,– DM	zzgl. Versandkosten	6,– DM
Endpreis	**73,– DM**	**Endpreis**	**75,– DM**

– Bitte benutzen Sie die Bestellkarte im Heft. –

DMV-Verlag · Postfach 250 · 3440 Eschwege DMV Daten- und Medienverlag



Tennis total. Mit Great Courts können Sie sowohl trainieren als auch Wettkämpfe ausfechten

Great Courts

Nicht erst seitdem Steffi die Weltranglisten-Beste ist, begeistert Tennis die Massen. Und wer es sich nicht leisten kann, im Tennisclub zu spielen, der besorgt sich eben ein Tennis-Computerspiel.

Mit Great Courts stellt sich ein neues Tennisspiel vor, das Sie in die Lage versetzt, an drei hochkarätigen Turnieren teilzunehmen, auch wenn Sie nicht in der Weltrangliste plaziert sind. Ihre Karriere beginnt in Melbourne, wo Sie versuchen müssen, das Australian Open für sich zu entscheiden. Ist dies gelungen, können Sie sich nacheinander für das French Open, All England Champion Ship und das U.S. Open qualifizieren. Wem das nötige Können fehlt, der sollte zunächst mit einer Ballwurfmaschine üben. Ebenfalls eine Option, die in Great Courts mit sechs unterschiedlichen Trainingsprogrammen enthalten ist.

Tennissimulationen haben es nicht leicht, und sei es nur deswegen, weil es so viele davon gibt. So muß es sich auch jedes neue Produkt gefallen lassen, an dem, was es bisher auf dem Markt gibt, gemessen zu werden. Ein Vergleich, den Great Courts nicht zu scheuen braucht. Grafik und Animation sind gut gelungen und werden von digitalisierten Soundeffekten, wie z.B. die Rufe des Schiedsrichters oder dem Aufschlag des Balles, glänzend untermalt. Daneben ist die Bedienungsanleitung zu loben, die sich nicht damit begnügt, dem Spieler zu erklären, wie das Spiel bedient wird, sondern gleich noch einen kleinen Abriß über Geschichte und Technik des Tennissportes liefert.

(hs)

AMIGA DOS Blitzlicht

Name: Great Courts
Hersteller: Blue Byte
Quelle: Fachhandel
Preis: 89,- DM

Positiv:
- gute Anleitung
- auch Training möglich
- verschiedene Schwierigkeitsgrade

Negativ:
- Zwei-Spieler-Option mangelhaft





Begleiten Sie den Helden mit seinem Freund, dem Drachen, bei seiner Suche nach den magischen Gegenständen. Achten Sie jedoch auf das Drachensymbol. Wenn nur noch ein Skelett zu sehen ist, ist die Lebensenergie verbraucht

Dragonscape

'Change into a strange world' war unser erster Gedanke, als wir des vorliegenden Softwarepaketes habhaft wurden. Um es vorwegzunehmen: Hier war der Wunsch der Vater des Gedanken.

Nach einer ungewöhnlich langen Ladezeit präsentiert sich Dragonscape mit einem Grafikfenster in der Mitte des Bildschirms. Dort sind der Drache, Sie selbst sowie das darunterliegende Szenario zu sehen. Der erste Level mit dem Wasteland gibt schon eine ungefähre Vorstellung des wüsten und öden Geländes, in dem Ihre Aufgabe ihren Anfang nimmt. Insgesamt acht Gegenstände sind großflächig über das Szenario verteilt. Diese müssen gefunden, aufgenommen und an der richtigen Stelle wieder abgelegt werden. Hört sich einfach an, gestaltet sich in der Praxis aber recht problematisch. Zahlreiche Feinde, von denen eine nicht unerhebliche Anzahl gegen Drachenfeuer sozusagen immun ist, versuchen ständig, Ihnen das Leben zu erschweren. Jede Berührung mit einem der Aliens kostet Sie eine gewisse Menge Lebensenergie, die allerdings nicht wieder aufgefrischt werden kann. Die Lebensenergie wird übrigens durch einen Drachen am rechten Bildrand angezeigt. Je mehr der Drachen skelettiert, desto geringer ist Ihre aktuelle Lebensenergie. Ein weiteres Fenster am unteren Bildrand zeigt an, welchen der im Level befindlichen acht Gegenstände Sie momentan gerade mit sich führen. Ein kleiner Balken informiert darüber, wie viele der Gegenstände Sie aufgenommen und plaziert haben.
Alles in allem ein fades und freudloses Game.

(mm)

AMIGA DOS Blitzlicht

Name: Dragonscape
Hersteller: Software Horizons LTD
Quelle: Fachhandel
Preis: 60,- DM

Positiv:
- passable Grafik

Negativ:
- absurder Hintergrund
- lange Ladezeiten
- geringe Motivation
- nervtötender Sound





X-OUT, Balleraction unter Wasser. Hier gilt es, die Aliens per U-Boot zu jagen



Ein Adventure aus der Zeit des kalten Krieges: Berlin 1948

Demnächst auf Ihrem Computer

Neue Spiele braucht das Land! Das Unterhaltungsgeschäft ist ein hartes Brot. Da macht das Geschäft mit der Computerunterhaltung keine Ausnahme. Jedes Unternehmen sucht DIE neue Spielidee. Auch wenn es nicht die ultimativen neuen Spielkonzepte sind, die wir Ihnen in dieser Preview vorstellen können, erwarteten Sie doch einige vielversprechende Neuheiten.

X-OUT

Das Wettrüsten geht weiter. Ein neues, actionreiches Ballerspiel steht kurz vor der Fertigstellung. Doch diesmal muß nicht der Sternenhimmel als Schauplatz herhalten, sondern die Tiefen des Meeres. Neben guter Musik und feiner Grafik besticht X-OUT ferner durch weitere kleine Besonderheiten. Über 25 verschiedene Extrawaffen, acht verschiedene Level und Extrawaffen, die nach eigenen Vorstellungen konfiguriert werden können, erwarten Sie.

Berlin 1948

Im Jahre 1948 blockierten sowjetische Truppen jeden Landzugang von und nach Berlin. Fortan konnte die Stadt nur noch mit Flugzeugen versorgt werden. Die Luftbrücke der westlichen Alliierten rettete die Stadt. Die Weltlage ist zum Zerreißen gespannt, und mitten in diesen politischen Hexenkessel platzt die Nachricht, daß den Amerikanern eine Atombombe abhanden gekommen ist. Berlin 1948 ist ein Spionageabenteuer etwas anderer Machart, denn ganz Berlin muß nach der Bombe abgesucht werden.

Loom

Sie mögen Märchen und Fantasie? Sie erleben gerne Abenteuer? Das trifft sich hervorragend, denn in wenigen Wochen wird Loom erscheinen, und wenn Sie beide Fragen mit 'Ja' beantwortet haben, dann ist Loom das Spiel für Sie. Eine tolle, stilvoll und stimmig erzählte Geschichte mit viel Magie, grimmigen Bösewichtern und verzauberten Damen. Interessanteste Neuerung sind die Befehle, durch die Loom kontrolliert wird. Keine Texte, keine Menüs, keine Icons. Magie heißt das Zauberwort – lassen Sie sich überraschen. Entworfen wurde dieses Spiel von Brian Moriarty, der sich schon mit der Zork-Serie einen Namen als führender Spieledesigner gemacht hat.

Heroes Quest

Sierra-Fans dürfen sich freuen. Heroes Quest, das neue Storyadventure aus diesem Hause, wird auch für den Amiga erscheinen. Diesesmal hat es den Sierra-Jungs nicht gereicht, ein einfaches Abenteuer zu gestalten. Sie haben auch noch eine gehörige Portion Rollenspiel mit hineingepackt. Sie können sich wieder auf viele Gags und harte Kopfnüsse in bester Sierramanier gefaßt machen. Ansonsten hat sich das Spielgeschehen durch die neuen Rollenspielelemente sehr zum Positiven gemausert.

(hs)



Magie und Fantasie – Loom wird Sie in eine zauberhafte Welt entführen



Sierras erstes Rollenspiel – Heroes Quest

Impressum

Herausgeber
Christian Widuch

Chefredaktion
Markus Matejka (mm)

Redaktion
Jürgen Borngießer (jb), Bernhard Rinke (br), Heinrich Stiller (hs), Joachim Freiburg (jf), Martin Schlöter (ms)

Freie Autoren dieser Ausgabe
Edgar Meyzis, Bernd Rudolf, Robert Marz, Jürgen Seibel, Michael Winterberg, Klaus Rupp, Garry Glendown, Ralph Conway, Ulf Peterson, Achim Stindt, Michael Trebing, Michael Anton, Frank Schiefer

Redaktionsassistenz
Anke Kerstan (ke), Susanne Eska (es)

Koordination
Stefan Ritter

Produktionsleitung
Gerd Köberich

Bereichsleitung
Claudia Ebbrecht (Fotosatz/Lektorat), Christian Heckmann (Fotografie/Layout), Margarete Schenk, Helmut Skoupy (Montage/Reprografie)

Layout
Yvonne Hendricks

Fotografie
Klaus Jatho, Heike Reifer

Fotosatz
Gabriela Joseph

Lektorat
Dagmar Wilhelm-Ballhaus, Angela Palm

Montage/Reprografie
Manuela Eska, Dieter Schnobl

Werbegestaltung
Mohamed Hawa, Petra Küch

Anzeigenverkaufsleitung
Wolfgang Schnell

Anzeigenverkauf
DMV-Verlagsbüro München
Zaunkönigweg 2c, 8000 München 82
Telefon (0 89) 4 39 10 87, Telefax 0 89/4 39 10 80
Leitung: Britta Fiebig
Anzeigenverkauf: Monika Schöbel, Jens Dhein, Peter Schätzle

Anzeigenpreise
Es gilt die Anzeigenpreisliste Nr. 1 vom 01.01.1990

Anschrift Verlag/Redaktion:
DMV Daten & Medien-Verlag
Widuch GmbH & Co. KG
Fuldaer Straße 6
3440 Eschwege, Telefon (0 56 51) 8 09-0,
Telefax (0 56 51) 8 09-3 33

Vertrieb
Verlagsunion Erich Pabel-Arthur Moewig KG (VPM), Friedrich-Bergius-Straße 20, 6200 Wiesbaden

Druck
Druckerei Jungfer, 3420 Herzberg

Bezugspreise
»AMIGA DOS« erscheint monatlich.
Einzelpreis DM 6,50/sfr. 6,50/öS 52,–

Abonnementpreise
Die Preise verstehen sich grundsätzlich einschließlich Porto und Verpackung.

Inland:
12 Ausgaben: DM 70,–
6 Ausgaben: DM 35,–

Europäisches Ausland:
12 Ausgaben: DM 100,–
6 Ausgaben: DM 50,–

Außereuropäisches Ausland:
12 Ausgaben: DM 120,–
6 Ausgaben: DM 60,–

Bankverbindungen:
Postscheck Frankfurt/M: Kto.-Nr.: 23043-608
Raiffeisenbank Eschwege:
BLZ: 522 603 85, Kto.-Nr.: 245 7008

Die Abonnementbestellung kann innerhalb einer Woche nach Auftrag beim DMV-Verlag, Postfach 250, 3440 Eschwege, schriftlich widerrufen werden. Zur Wahrung der Frist reicht der Poststempel. Das Abonnement verlängert sich automatisch um 6 bzw. 12 Monate, wenn es nicht mindestens 6 Wochen vor Ablauf beim Verlag schriftlich gekündigt wird.
Für unverlangt eingesandte Manuskripte und Datenträger sowie Fotos übernimmt der Verlag keine Haftung. Die Zustimmung zum Abdruck wird vorausgesetzt.
Eine Haftung für die Richtigkeit der Veröffentlichungen kann trotz sorgfältiger Prüfung durch die Redaktion vom Herausgeber nicht übernommen werden. Die geltenden gesetzlichen und postalischen Bestimmungen sind zu beachten.

Die gewerbliche Nutzung, insbesondere der Programme, Schaltpläne und gedruckten Schaltungen, ist nur mit schriftlicher Genehmigung des Herausgebers zulässig. Das Urheberrecht für veröffentlichte Manuskripte liegt ausschließlich beim Verlag. Nachdruck sowie Vervielfältigung oder sonstige Verwertung von Texten nur mit schriftlicher Genehmigung des Verlages.
Namentlich gekennzeichnete Fremdbeiträge geben nicht in jedem Fall die Meinung der Redaktion wieder.

Die Inserenten

AMIGA DOS

Alle Programme sofort nutzen

Da ist er nun endlich – der Trick oder das Programm, auf das Sie schon so lange gewartet haben! Zu allem Unglück ist das Listing aber in Modula-2 oder C, jedenfalls in einer Sprache, zu der Sie keinen Compiler haben, um ein lauffähiges Programm herzustellen.

Auch in diesem Fall hilft Ihnen die Databox von Amiga DOS aus der Patsche: Neben den Quelltexten im ASCII-Format finden Sie jeweils auch das fertige, lauffähige Programm. Sie brauchen es nur von der Databox-Diskette aus zu starten.

Endlich keine Listings mehr abtippen!

Nicht bei allen Programmen ist es mit drei Zeilen getan – gute Routinen und praktische Funktionen brauchen ihren Platz. Und bisweilen lassen sich auch lange Datenblöcke nicht vermeiden, ganz zu schweigen von Hexdumps und Assemblerlistings.

Schonen Sie Ihre Augen und schlagen Sie sich nicht die Nacht mit Abtippen um die Ohren. – Auf der Databox zum Amiga DOS-Heft finden Sie alle Listings als ASCII-File: passend für jeden Texteditor, den Amiga-BASIC-Interpreter, Makro-Assembler oder einen Compiler für C und Modula-2.

DATA

Alle Listings und Programme auf Diskette –

Sofort anfangen – ohne Zeitverlust!

Für was Sie sich auch interessieren, Interna des Amiga, Anwendungen, oder die Beispielprogramme der Werkstätten im Amiga DOS-Heft: Bevor Sie richtig an die Arbeit gehen können müssen Sie erst die benötigten Programme eingeben, korrigieren und zum Laufen bringen.

Sparen Sie sich diesen ermüdenden, unnötigen Aufwand an Zeit und Konzentration. Mit der Databox für Ihr Amiga DOS-Heft können Sie sofort den Dingen auf den Grund gehen, die Sie wirklich interesssieren.

Keinen Ärger mehr mit Tippfehlern

Wer kennt das nicht, wenn das Programm nach dem Eintippen nicht läuft oder der Rechner gar abstürzt. Besonders gemein sind auch Fehler, die erst nach Wochen bei einer bisher nicht gebrauchten Funktion zu Tage treten, oder wenn der Druckfehlerteufel am Werke war.

Zermartern Sie sich nicht den Kopf, bis Sie die falsche Zahl im Datafeld gefunden haben. – Alle Dateien auf der Databox zur Amiga DOS sind vom Autor und der Redaktion auf Fehlerfreiheit geprüft und im dazugehörigen System "probegelaufen".

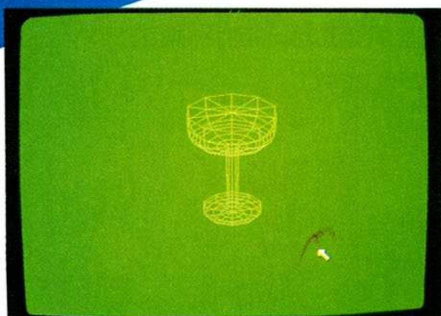


GuruOff – Jetzt ist Schluß mit unverständlichen Meditationen. Wir holen mit unserem Programm den Guru von seiner Wolke

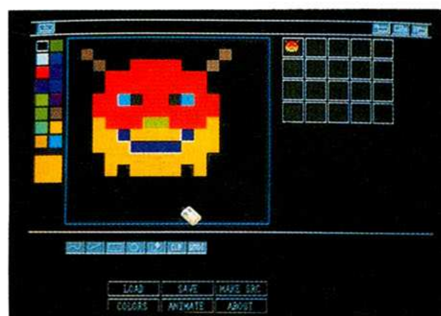


BOX

Computer einschalten – Diskette einlegen – los geht's



Aus 2D wird 3D – ROD, das Programm zur Berechnung und Darstellung von Rotationskörpern, macht's möglich



Das Utility zur Eingabe von BOBs, Sprites und Images. SprEd, unser Sprite-Editor-Projekt, macht das Erstellen dieser beweglichen Objekte zum Kinderspiel

Inhalt der Databox Amiga DOS 2/90

Basic	C	Modula-2	Assembler	Lauffähiges Programm	
x				x	BASIC-Kurs – alle Beispielprogramme lauffähig auf der Databox
				x	ConText AMIGA lernt drucken – Die Druckroutinen für 24- und 9-Nadel-Drucker in GFA-Basic
	x				Das Sprite-Editor-Projekt – Der erste Teil zur professionellen Erstellung von Sprites, BOBs und Images
				x	Virtest: Das Listing aus Ausgabe 1/90 zusätzlich als Bonusprogramm. Wir geben Viren keine Chance
		x		x	Rotationskörper im Eigenbau. ROD wandelt zweidimensionale Grafiken in dreidimensionale Körper um. Mit Editor!
			x	x	Der zweite Teil des Musikkurses – Musikstücke per Interrupt steuern
			x	x	GuruOff – Gurumeditationen mit neuem und verständlichem Requester. Es ist stellenweise sogar möglich, den Task-Held zu umgehen
			x	x	Beispielprogramme zur Blitter- und Copperprogrammierung
			x	x	Aufsteigerwissen Assembler: Paintbox V. 0.2, der zweite Teil unseres Grafikprogramms

24,– DM

Wenn Sie über den DMV-Bestellservice bestellen, gilt folgendes:

Inland:		**Ausland:**	
Einzelpreis	24,– DM	Einzelpreis	24,– DM
zzgl. Versandkosten	4,– DM	zzgl. Versandkosten	6,– DM
Endpreis	**28,– DM**	**Endpreis**	**30,– DM**

Zahlungsweise:
Am einfachsten per Vorkasse (Verrechnungsscheck) oder als Nachnahme zuzüglich der Nachnahmegebühr. (Bei Lieferungen in das Ausland ist Nachnahme nicht möglich.)

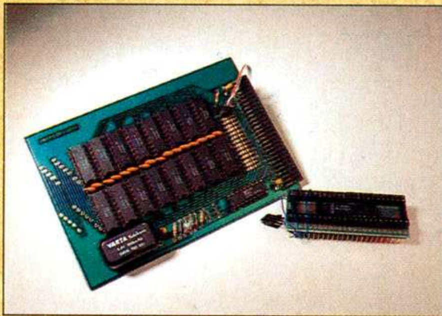
Bitte benutzen Sie die Bestellkarte.

DMV-Verlag · Postfach 250 · 3440 Eschwege



Im nächsten Heft

■■■ Speicher satt Speichererweiterungen für den Amiga sind in vielen Fällen unumgänglich, wenn man mit größeren Anwendungsprogrammen arbeitet. Das Angebot ist Legion – wo bekommt man die größte Leistung fürs Geld?



Speichererweiterungen für den AMIGA – viel Leistung für wenig Geld

■■■ Viruswürger Viren sind mit die ärgsten Feinde der Amiganer. Das muß nicht sein! Denn inzwischen gibt es genügend Viruswürger, mit denen sich der geplagte Anwender zur Wehr setzen kann. Eine Übersicht zeigt, wie Viren effektiv zu bekämpfen sind.

■■■ Im Test Turbo Print II – ein neues Druckerutility für den Amiga. Ob sauberes Drucken in Farbe oder schwarzweiß mit dem Laserdrucker – ist Turbo Print II die Lösung für Sie? Näheres in der nächsten AMIGA DOS.

■■■ Spiele Als Fensterputzer können Sie was erleben. Und damit sind nicht nur die diskreten Einblicke gemeint, denen man als Fensterputzer kaum ausweichen kann…
Lesen Sie in der nächsten Ausgabe von AMIGA DOS den Testbericht über Window Wizard.

■■■ Hardwarebastelei Wer kennt nicht das Problem, daß im gleichen Port Maus und Joystick vorhanden sein sollten. In der nächsten Ausgabe finden Sie eine Bauanleitung für einen Umschalter. So werden Ports und Nerven nachhaltig geschont.

■■■ Helpline Haben Sie Probleme mit dem einen oder anderen Spiel? Das muß nicht sein. AMIGA DOS stellt jeden Monat Tips zu den beliebtesten Spielen zusammen. In der nächsten Ausgabe finden Sie unter anderem Tips zu "Indiana Jones".

■■■ Und natürlich wieder jede Menge Tips&Tricks zu AmigaBASIC, ASSEMBLER, Modula2 und C!



Virenbekämpfung auf einen Blick

Die nächste



finden Sie ab

14. Februar '90

bei Ihrem Zeitschriftenhändler



Fensterputzen mit Schwierigkeiten. Nur ein echter "Window Wizard" kann es hier zur Meisterschaft bringen

STEFAN OSSOWSKI'S **SCHATZTRUHE**

(1) **Haushaltsbuch** bis zu 25 Konten, flexibel, leicht bedienbar, mit umfangreicher **deutschsprachiger** Dokumentation DM 8,-

(3) **MountainCAD** professionelles CAD-Programm, **deutsche Anleitung** DM 8,-

(4) **Spiele I, II, III** 10 erstklassige PD-Spiele aus allen Bereichen wie Action, Geschicklichkeit, Strategie (3 Disketten) DM 24,-

(5) **Anti-Virus** 8 Programme gegen alle Viren DM 8,-

(6) **Text** hochwertige **deutsche** Textverarbeitung DM 8,-

(7) **Utility-Disk** 25 nützliche Utilities aus allen Bereichen DM 8,-

(9) **Sonix-Paket** Original-Sonix-Player + 4 weitere Disketten mit phantastischer Sonix-Musik. **Top-Hit!** DM 40,-

(10) **Business** 3 Disketten: Tabellenkalkulation, relat. Datenbank, sehr gute Textverarb. Vers. engl. DM 24,- Vers. deutsch DM 70,-

(13) **Paranoid** sensationelles Breakout-Spiel DM 8,-

(14) **Buchhaltung** erstes **deutsches** PD-Buchhaltungsprogr. DM 8,-

(16) **AMIGA-Paint** sehr gutes **deutsches** Malprogramm DM 8,-

(17) **Videodatei** bringt Ordnung in Ihre Videodatei, **deutsch** DM 8,-

(18) **Fußballmanager** bei diesem Spiel können Sie Ihre Fähigkeiten als Manager eines Fußballclubs testen, **deutsch** DM 8,-

(19) **Platten/Cassetten/CD-Liste, deutsch** DM 8,-

(20) **Giroman** komfortables **deutsches** Programm, mit dem Sie Ihr Girokonto einfach verwalten können DM 8,-

(22) **Kampf um Eriador, V 2.0** taktisches Strategiespiel für 2 Personen mit sehr guter Grafik und Sound, **deutsch** DM 8,-

(25) **Label 2.0** Etikettendruckprogramm, **deutsch** DM 8,-

(26) **Risiko** die Amiga-Umsetz. d. bek. Brettspiels, **deutsch** DM 8,-

(27) **Ray-Tracing-Construction-Set, V 2.0** phantastisches Programm zur Berechnung von Licht und Schatten - siehe Test Amiga 1/88 - komplett auf **3** Disketten mit **deutscher Anleitung** DM 24,-

(28) **Wizard of Sound** ein phantastisches Musikprogramm zur Erstellung eigener Lieder, mit **deutscher** Anleitung (2 Disks) DM 10,-

(29) **Broker** ein sehr gutes **deutsches** Börsenspiel DM 8,-

(30) **Quickmenü** erst. Sie sich Ihre eig. Workbench i. **deutsch** DM 8,-

(31) **Blizzard** phantastisches Ballerspiel m. sehr guter Animat. DM 8,-

(32) **DSort deutsches** Diskettenkatalogisierungsprogramm DM 8,-

(33) **Pascal** ein komplettes Pascal-Paket (3 Disketten) mit Compiler, **deutscher Anleitung** u. einem s. gut. deutschen Editor DM 24,-

(34) **DiskKey** Diskettenmonitor mit **deutscher Anleitung** DM 8,-

(35) **Peters Quest** Geschicklichkeitsspiel mit lustiger Handlung und **deutscher Anleitung** DM 8,-

(36) **Spiele** auf dieser Diskette sind 3 Tetris-ähnl. Spiele enth. DM 8,-

(37) **MRBackup** Festplattensicherungsprogramm mit **deutscher Anleitung** DM 8,-

(38) **Universal-Datei deutsches** Datei-Verwaltungsprogr. DM 8,-

(39) **Assembler** ein komplettes Entwicklungssystem für Maschinensprache in **deutsch!** DM 8,-

(40) **Bibel-Quiz** lehrreich und unterhaltsam DM 8,-

(41) **Faktura** Fakturierungsprogramm inkl. Mahnungen und Adressen – mindestens 1 MB Speicher DM 8,-

(42) **BootMaster** Mit diesem Programm können Sie individuelle Bootblöcke mit Lauftext und Sternenhintergrund erstellen. DM 10,-

(43) **Banner II** Dieses Programm ermöglicht es Ihnen, komplette Banner mit Ihrem Drucker zu erstellen. Leicht bedienbar! DM 8,-

*** TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN ***

Die deutsche Public-Domain-Serie aus dem Hause Ossowski! Wußten Sie schon, daß beim PD-Versand Stefan Ossowski alle drei Monate 10 neue TAIFUN-Disketten mit den interessantesten Neuerscheinungen des PD-Marktes vorgestellt werden? Bevorzugt präsentieren wir Ihnen die neuesten deutschen Programme. Am 10. November erschienen die neuen TAIFUN-Disketten Nr. 111 bis 120.

Schnupperpreis: DM 53,- V-Scheck, DM 57,- Nachnahme

Professionelle Anwendungssoftware für den anspruchsvollen User:

(100) **DME T & W = Text & Wörterbuch**
DME T & W ist ein leistungsstarker Texteditor sowohl für den Viel- als auch für den Gelegenheitsschreiber. DME T & W ist äußerst flexibel und sogar programmierbar. Sie erhalten DME T & W bei uns mit **deutscher Anleitung** und einem eingebauten **Deutsch-Englisch-Wörterbuch** mit ca. 3000 Vokabeln. DM 30,-
DME mit englischer Anleitung und ohne Wörterbuch: DM 6,-

(101) **RIM-5 = Relationale Datenbank**
RIM-5 ist eine äußerst leistungsfähige relationale Datenbank, die sowohl für den geschäftlichen als für den privaten Bereich prädestiniert ist. Natürlich liefern wir RIM-5 mit **deutscher Anleitung** und einem **ausführlichen Einführungskurs**. DM 30,-

(102) **AnalytiCalc = Tabellenkalkulation**
Eine leistungsstarke Tabellenkalkulation mit **deutscher Anleitung**. Schon nach kurzer Zeit wird AnalytiCalc ein unverzichtbares Hilfsmittel für Kalkulationsaufgaben im Büro oder zu Hause sein. DM 30,-

(103) **DEA Arithmetica = Die Göttin Arithmetica**
DEA Arithmetica ist ein Programm, auf das Schüler, Studenten und Lehrer schon lange gewartet haben. Ableitungen, Kurvendiskussionen und das Skizzieren von Funktionsgraphen stellen von nun an kein Problem mehr dar. **Made in Germany!** DM 30,-

(104) **Haushaltsbuch Version 2.0**
Die Komplettlösung zur Verwaltung Ihrer privaten Finanzen. Wollen Sie einen Überblick über Ihre Finanzen, Ihr Geld effektiver nutzen und Schwachstellen in Ihrer Budgetierung erkennen? Haushaltsbuch V. 2.0 ist leicht bedienbar und auch für Computerneulinge hervorragend geeignet! Demo-Dateien unterstützen sie bei Ihren ersten Schritten und das Programm ist frei von buchhalterischen Fachausdrücken. Ein Funktionsüberblick: Führen von verschiedenen Dateien, mehrere Kontenlisten, bis zu 10 Bilanzen, 40 freidefinierbare Konten, Suchroutinen, viele Voreinstellungen, doppelte Buchführung, Datei-Ex- und -Import, Filterfunktionen, Jahresübertrag, Mausunterstützung ... **Selbstverständlich mit umfangreicher deutscher Dokumentation!** Haushaltsbuch V 2.0 benötigt 1 MB Speicher. DM 98,-

(105) **Xytronic II**
Bei diesem Spiel können Sie intergalaktischen Handel betreiben. Um Ihren Reichtum zu mehren, müssen Sie geschickt kaufen und verkaufen und natürlich den Weltraumpiraten aus dem Weg gehen. Sehr gute Grafik und sehr guter Sound garantieren zusätzlich eine hohe Motivation. DM 29,-

(106) **Data-Manager**
Eine sehr leicht zu bedienende Dateiverwaltung, mit der Sie Adressen, Mitgliederlisten etc. perfekt verwalten können. Ein **deutsches Programm** mit **deutscher Anleitung**. DM 15,-

(107) **EGOS = European Game of Strategy**
Ein Spiel für 2 Spieler. Sie kämpfen um die Vorherrschaft in Europa. Versuchen Sie, die Taktik Ihres Gegners zu durchschauen und Ihre eigene Strategie entsprechend auszurichten. Ein Muß für Strategiefans. **Deutsch!** DM 29,-

(108) **Bundesliga 2000**
Ein leistungsstarkes Programm zur Bundesliga-Ergebnisverwaltung. Auf einer Diskette können Sie bis zu 40 Ligen verwalten. Mit Historie seit 1985. **Komplett in Deutsch!** DM 15,-

(109) **Money Player Deluxe – Geldspielgerät**
Sie fühlen sich wie in der Spielhalle! Start- und Risikoautomatik, Sonderspiele, viele Extras und ein toller Spielspaß. Spielkomfort durch Maussteuerung, Pal-Auflösung ... DM 39,-

(110) **Wizard of Sound 2.0**
WoS ist ein phantastisches Musik- und Komponierprogramm. Erstellen Sie eigene Lieder entweder durch Noten und per Klaviatur. **WoS** verfügt über 60 verschiedene Instrumente, einen Instant-Replay-Modus und eine große Funktionsvielfalt. **WoS** wurde **in Deutschland entwickelt** und wird mit **umfangreicher deutscher Dokumentation** komplett auf **2 Disketten** geliefert! Inkl. Demosongs und Player DM 35,-

(111) **Broker 2.0 + Das Börsenprogramm**
Broker 2.0 ist eine sehr **realistische Börsensimulation**. Alleine oder mit bis zu 3 Mitspielern versuchen sie Ihr Kapital zu mehren. Nur einer kann gewinnen! Dieses Spiel ist unterhaltsam, lehrreich, zeichnet sich durch lang anhaltende Motivation aus – 1MB Speicher DM 29,-

(112) **Vokabel-Trainer-English 1.0**
Der **Vokabel-Trainer** ist **ein einfach zu bedienendes, leistungsfähiges Lernprogramm**. Zusätzlich zu den bereits vorhandenen Vokabeln kann man den Wortschatz leicht erweitern. ein Programm, mit dem das Lernen Spaß macht! **Made in Germany!** DM 15,-

(113) **Pente**
Ein »**5-Gewinnt-Spiel**«, erweitert um einige Regeln. Sie spielen gegen den Computer. Für jung und alt mit hoher Motivation DM 19,-

Die Programme 100 bis 103, 105 bis 108 und 110 bis 113 erhalten Sie in dieser Ausführung ausschließlich beim PD-Vertrieb Stefan Ossowski!

Versandkosten	**Inland:**	**DM 3,- V-Scheck DM 7,- Nachn.**
(Porto/Verpackung):	**Ausland:**	**DM 6,- V-Scheck DM 15,- Nachn.**

ABO-SERVICE
Bei uns erhalten Sie fast jede PD-Serie auch im günstigen Abonnement! Auf unsere Staffelpreise gewähren wir außerdem einen **10%igen ABO-Rabatt**! Die Fish-Serie ist z. B. schon bis Nr. 260 lieferbar! Rufen Sie uns doch einfach an oder schreiben Sie uns, wenn Sie an weiteren Informationen zu unserem ABO-Service interessiert sind!

Zuverlässigkeit
\+ Schnelligkeit
\+ Service
= PD-Versand Stefan Ossowski

Testen Sie uns!

Wir führen alle bekannten PD-Serien wie Fish, Panorama, Faug Amicus, Auge, Taifun, Chiron, RPD, Kickstart, Slideshows, TBAG, Franz ... Viele Programme deutschsprachig.

5,-DM kosten unsere aktuellen **2** Katalogdisketten. Mit deutschem Inhaltsverzeichnis unseres gesamten PD-Angebots von weit über 1000 Disketten. Ab DM **4,50**. Gegen Verrechnungsscheck oder in Briefmarken anfordern.

Stefan Ossowski - Ihr PD-Spezialist - Veronikastr. 33, D-4300 Essen 1, Tel./Btx: 0201/788778

Die Wüste lebt

IT CAME FROM THE DESERT™

SEE DECENT AMERICANS TERRORIZED!

HEAR TEENAGE GIRLS SQUEAL!

PRESENTED IN THRILL-O-VISION!

MAKE THEM ANTS CRY UNCLE!

Wer Lizard Breath kennt, käme nie auf den Gedanken, daß irgendwann einmal etwas Schreckliches die beschauliche Ruhe des kleinen Örtchens am Rande der Wüste stören könnte. Dann kommt der Tag, an dem ein Meteorit in der Wüste niedergeht und mutierte Insekten die kleine Stadt bedrohen. IT CAME FROM THE DESERT – ein Horrorfilm im Stil der 50er Jahre, von CINEMAWARE für Ihren AMIGA in Szene gesetzt.

Versionen für Atari ST und PC folgen.





CINEMAWARE
Mirrorsoft

Ariola Soft
Das Programm